Quantitative Approaches to enable the Automated Planning of Adaptive Process Models

Universität Regensburg

DISSERTATION
zur Erlangung des Grades
eines Doktors der Wirtschaftswissenschaft

eingereicht an der
Fakultät für Wirtschaftswissenschaften
der Universität Regensburg

vorgelegt von
Dominik Schön, M.Sc.

Berichterstatter:
Prof. Dr. Bernd Heinrich
Prof. Dr. Mathias Klier

Tag der Disputation:
23.06.2020

Dedicated to my parents
Elisabeth and Helmut Schön,

my beloved wife Sarah,

and the memory of my grandparents
Elisabeth and Fritz Schön

Acknowledgements

This dissertation was a long, hard journey. Not only academically but especially personally. Thus, I want to express my gratitude to those who took part in this journey.

First to mention, my doctoral supervisor Prof. Dr. Bernd Heinrich. He has introduced me to the scientific world and taught me how to work scientifically - at least to some extent. I am deeply thankful for the trust he has placed in me. In times when I no longer believed in a successful end to this journey, he encouraged me. In (the more frequent) times when I've taken things a little too lightly, he challenged me.

Second, I want to thank Mathias Klier. As my co-supervisor he was not only a tough sparring partner but also a good coach and always stood by the side with advice and open ears - and he taught me how laundry facilities work.

Third, I want to thank my coworkers. Especially I want to thank Diana Hristova and Lars Lewerenz who shared the experience of starting a new journey in the scientific world combined with building up a new chair at the University of Regensburg in the first months of our professional career. Further, I want to thank my co-authors Alexander Schiller and Michael Szubartowicz for their fruitful contributions, ideas, feedback, and especially our different views on the topics that we worked on, together.

I want to thank my family for their ongoing support throughout the last years. With the greatest admiration I thank my parents. They have supported me since childhood and with full power. Life has always been easy for me, because my parents were always there for me when I had worries - or didn't even let me have any. Thank you, mom and dad.

Finally I want to thank my wife Sarah. Without her, I probably would not have started this dissertation and certainly would not have finished it. Especially when starting this dissertation, I oftentimes had doubts whether I had what it takes for this journey. She always stood behind me in such situations and put her own needs in the background. She encouraged me to believe in myself, she was my personal motivator, when I was going through slumps. Thank you, Sarah, for always supporting me. I love you.

Table of Contents

List of Abbreviations

AI	Artificial Intelligence
BPEL	Business Process Execution Language
BPM	Business Process Management
BPMN	Business Process Model and Notation
EPC	Event-driven Process Chains
IoT	Internet of Things
OWL	Web Ontology Language
PAIS	Process Aware Information System
QoS	Quality of Service
RQ	Research question
SBPM	Semantic Business Process Management
SME	Small and Medium Enterprise
SOA	Service-Oriented Architecture
UML	Unified Modeling Language
WFMS	Workflow Management System

List of Figures

1 Introduction

2 Automated Planning of Adaptive Process Models

Paper 1: Automated Planning of Process Models: The Construction of Simple Merges

Paper 2: Automated Planning of context-aware Process Models

Paper 3: The Cooperation of Multiple Actors within Process Models: An Automated Planning Approach

3 Automated Adaptation of Existing Process Models

Paper 4: Adapting Process Models via an Automated Planning Approach

4 Evaluation of Automated Planning with respect to the Efficiency of Process Modelers

Paper 5: The Influence of Automated Planning on the Task Performance of Process Modelers

7 Appendices

List of Tables

1 Introduction

2 Automated Planning of Adaptive Process Models

Paper 1: Automated Planning of Process Models: The Construction of Simple Merges

Paper 2: Automated Planning of context-aware Process Models

Paper 3: The Cooperation of Multiple Actors within Process Models: An Automated Planning Approach

3 Automated Adaptation of Existing Process Models

Paper 4: Adapting Process Models via an Automated Planning Approach

4 Evaluation of Automated Planning with respect to the Efficiency of Process Modelers

Paper 5: The Influence of Automated Planning on the Task Performance of Process Modelers

7 Appendices

1 Introduction

Explanatory note: The references for the papers are listed at the end of each paper. The references for the introduction and the conclusion are at the end of the according chapters. Additionally all references used in the whole manuscript are listed at the end in Chapter 6.

For consistency reasons, the appendices of all papers of this dissertation have been moved to the end of this manuscript (see Chapter 7).

Processes serve as a central nervous system for today's business world that "changes faster all the time" (Harmon, 2019, p. 441). Companies deliver products or services by carrying out processes. Davenport (1993, p. 5) defines a process as "a structured, measured set of activities designed to produce a specific output for a particular customer or market. It implies a strong emphasis on how work is done within an organization [...]. A process is thus a specific ordering of tasks across time and space, with a beginning and an end, and clearly defined inputs and outputs: a structure for action. [...] Processes are the structure by which an organization does what is necessary to produce value for its customers." This follows the way of thinking of Taylor (1911) who laid the foundation for modern management principles: "Perhaps the most prominent single element in modem scientific management is the task idea. [...] This task specifies not only what is to be done but how it is to be done" (p. 39). Processes evolved from individual tasks (i.e., activities in the nomenclature of Davenport and referred to as actions in the remainder) by combining them together. To illustrate the focus of this dissertation, the following, simplified example of processes of car manufacturers will be used. It serves as an appropriate example to frame the focus of the dissertation at hand. Initially, before constructing a car, a car manufacturing company designs it. For this, product designers, technicians and engineers work together to define concepts and finally build prototypes. After a prototype has been built and several pretests have been conducted, the batch production can be started. To do so, car manufacturers usually work together with a network of suppliers in the supply chain that conduct parts of processes (e.g., prefabricating the cables, axes, seats, and plastics of the car). The car manufacturer thereafter assembles the car and delivers the car to the according customer. Thereby, "processes generate most of the costs of any business" as well as "strongly influence the quality of the product and the satisfaction of the customer" (Powell et al., 2001, p. 64).

Business Process Management (BPM) as a scientific and professional discipline emerged in the last decades (Mathiesen et al., 2011) and according to Bandara et al. (2009) has become a powerful competitive tool for organizations. BPM adoption has a positive impact on the organizational performance of the according companies (Bach et al., 2019) and even small and medium-sized enterprises (SME) are increasingly adopting BPM (cf., e.g., Braunnagel et al., 2016). Its definitions are manifold (cf. Rosemann et al., 2005), however the focus is often on analyzing and improving processes (Zairi, 1997). Armistead et al. (1997) consider BPM as a holistic approach that aims at managing processes on an ongoing basis. Process modeling as an important activity in the field of BPM comprises the construction of process models. Process models, depict processes as implemented and executed in reality (so-called "as-is" process models) or as they should be in a future implementation and execution (so-called "to-be" process models; Rosemann et al., 2015). Process models are usually (semi-) formalized in

a particular process modeling notation such as EPC (Event-driven Process Chains; cf. Keller et al. 1992; Nüttgens et al. 2002; Nüttgens et al. 1998), BPMN (cf. *Business Process Model and Notation (BPMN): Version 2.0.2* 2013) and UML (*OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015). The mentioned process modeling notations are referred to as imperative process modeling notations. Besides these, different modeling approaches such as declarative modeling exist (cf., e.g., Pesic et al., 2006; Pichler et al., 2011; Prescher et al., 2014; van der Aalst et al., 2009), but are not in the scope of the dissertation at hand. Structured, imperative process models comprise different modeling elements like actions, control flow patterns and edges. Control flow patterns depict the interdependencies of single actions of a process (Russell et al., 2006a; Russell et al., 2016; van der Aalst et al., 2003). Process modeling has proven to be a crucial instrument for decision-makers (Rosemann et al., 2015) and assists for the development of information systems (Aguilar-Savén, 2004; Mendling et al., 2012a), in business reorganization projects (Becker et al., 2010a; Mendling et al., 2010), and for communication and training purposes (Branco et al., 2014).

In the following Section, several challenges in the field of BPM are presented, which serve as a motivation for the dissertation at hand. Thereafter, based on these challenges, two particular aspects, which lay the foundation for this dissertation will briefly be sketched in Sections 1.2 and 1.3. The particular focus of the dissertation at hand will be introduced in Section 1.4 and addressed in Chapters 2 to 4 before Chapter 5 concludes the dissertation with a brief summary of the major findings as well as an outlook to further research.

1.1 Challenges in BPM

Vom Brocke et al. (2014) provide a brief overview of works that discuss critical success factors of BPM. Most of them conclude by presenting lists of more or less general factors such as "top management support", "appropriate culture", or "end-user training". However, vom Brocke et al. (2014) propose ten principles for what they call "good Business Process Management". Bandara et al. (2007) conducted a qualitative study among 14 "renowned BPM experts across the globe" on major issues in BPM. They distinguish between strategic, tactical and operational issues. At the strategic level the lack of governance, employee buy-in, and common mind share of BPM as well as a "broken link between BPM efforts and organi[z]ational strategy" are mentioned. At the tactical level, they refer to the lack of standards, BPM education, and methodology as well as "weaknesses in process specification". In the field of operational issues, they mention the "lack of tool support for process visuali[z]ation", "perceived gaps between process

design and process execution", and "miscommunication of tool capabilities" as major challenges.

With respect to the ten principles proposed by vom Brocke et al. (2014), the principle "continuity" means that BPM should be a permanent practice and not a "one-off project" and "holism" means that BPM "should be inclusive in scope" and not isolated. In line with that, Bandara et al. (2007) state that there is oftentimes no connection between tools for designing and simulating processes and tools to conduct processes, which leads to "large amount of rework". Further, works analyzed by vom Brocke et al. (2014) propose the "principle of simplicity", which focusses on the fact that BPM should be economical and not over-engineered. Other works underline that: Within the research field of BPM, the "value of business process modeling" is understood as one of the leading challenges (cf. Indulska et al., 2009). In particular, a study among Slovenian and Croatian companies, conducted by Škrinjar et al. (2010), underlines this fact, as most BPM initiatives they examined, aim at increasing the performance of organizations by means of improving the underlying processes. Škrinjar et al. (2010) as well as Glavan et al. (2017), in addition to that, show that more process-oriented companies perform better than less process-oriented companies and according to Bandara et al. (2009) BPM has become a powerful competitive tool for organizations. In general, relying on processes and improving them, nowadays is an economic key success factor and increasing the value of BPM is a major challenge, today.

To increase the economic benefits that BPM creates, researchers and practitioners have two levers that could be applied. On the one hand, the value that is created by BPM initiatives could be increased. However, even if this possibility exists in theory, an increase of the created value is often not guaranteed when applying such BPM initiatives. On the other hand efforts for BPM initiatives could be decreased. In particular, BPM initiatives nowadays can be complex and time-consuming in practice (Bowers et al., 1995; Škrinjar et al., 2010). "BPM initiatives can easily be set up consuming enormous amounts of resources. The principle of simplicity suggests that the amount of resources (e.g.[,] effort, time, money) invested into BPM should be economical" (vom Brocke et al., 2014, p. 533). Researchers therefore propose that "BPM should make opportune use of technology" (vom Brocke et al., 2014, p. 533) to reduce effort. In particular, the construction of process models is time-consuming and thus costly as it is mainly executed manually in practice (cf. Hornung et al., 2007). In a survey conducted by Becker et al. (2010b) among 60 banks "over two thirds [...] have a negative effort-utility-ratio concerning their process modeling initiatives" (p. 52). Thus, reducing the necessary manual efforts during process modeling initiatives or BPM initiatives, in general, seems promising for increasing the economic benefits of BPM.

Besides that, with concepts like Lean Manufacturing (Shah et al., 2007) or Agile Soft-

ware Development (Beck et al., 2001), companies primary goal conclusively shifted from long-term planning periods to just-in-time productions and release at will. Just-in-time supply chains arose and customer centricity became an increasingly important driver for organizations. Since then, agility (Goldman et al., 1994) has probably become the most important success factor for modern companies (Harraf et al., 2015; Jin-Hai et al., 2003; Wu et al., 2017; Zhang et al., 2000). This shift not only requires BPM initiatives to be efficient and fast but also requires the underlying business processes of companies to be flexible. Organizations increasingly evolved to process-centric businesses. "Technology [shifted] from being a process driver to a process enabler" (Lusk et al., 2005, p. 3). Thus, processes have to be created and optimized on the fly and change is a major objective (Smith et al., 2003). Reichert et al. (2012) rely on the variety of processes in hospitals to highlight the wide range of flexibility required in processes. Organizational and administrative processes are usually highly standardized as well as repetitive and may be pre-specified on the one hand, while a fracture treatment process is highly individual, depending on the current patient, on the other hand. While many current BPM approaches are based on rigid process models, its application is potentially challenging in dynamic domains where a high degree of flexibility is required (cf., e.g., Marrella, 2019). Hence, a second major challenge that serves as a foundation for this dissertation is the demand for flexibility of processes.

To sum up, two major challenges lay the foundation for the remainder of this dissertation. On the one hand, increasing the value of BPM and thus the efficiency of BPM approaches is of particular interest with respect to an economic point of view. On the other hand, flexibility of processes is required due to an ever-increasing dynamic environment. The next Section will briefly outline different possibilities to (semi-)automate selected activities of BPM to increase the flexibility of processes and to decrease manual efforts. Thereafter, business process flexibility will be defined as a foundation before in Section 1.4 the focus of the dissertation at hand will be presented.

1.2 Business Process Automation

To increase the efficiency of processes and their management as well as the agility of companies and to decrease manual efforts, several approaches to automate activities in the field of BPM have been proposed in the last years. "Countless IT solutions can be used to foster the efficiency and effectiveness of business processes" (vom Brocke et al., 2014, p. 538) and BPM in general. They aim at supporting modelers and business analysts as well as automating the conduction of processes to reduce the required manual efforts, which is one of the major challenges in BPM research (cf., e.g., Indulska et al., 2009) as previously sketched in Section 1.1. Workflow management systems (WFMS)

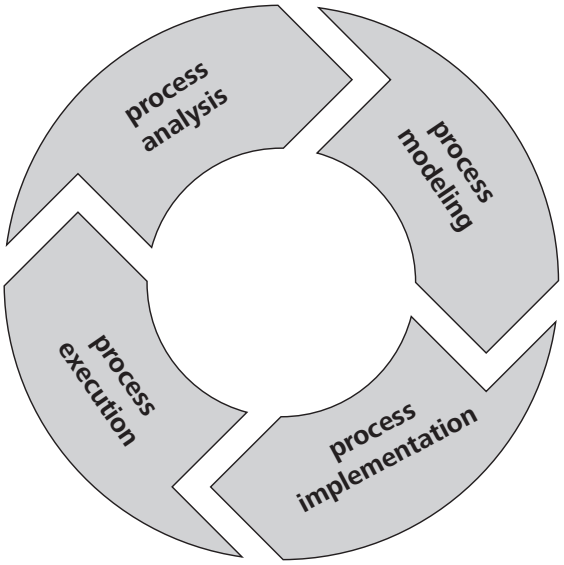

Figure 1.1: BPM Lifecycle proposed by Wetzstein et al. (2007)

or more generally process-aware information systems (PAIS), which are continuously developed further (cf., e.g., Pourmirza et al., 2019), lay a strong foundation for Business Process Automation. Especially through the rise of Service-Oriented Architectures (SOA), information systems are considered as a set of connected services (cf. van der Aalst, 2009). In this context, PAIS serve as the "glue" (van der Aalst, 2009), connecting the services. SOA serves as one of the major enabling technologies for emerging trends such as the Internet of Things (IoT) as it strongly relies on the core concepts of loosely coupled entities that share data, communicating with common standards (cf., e.g., Thakur et al., 2019). In the following, the previously introduced example of an automobile manufacturer will be used for sketching automated approaches in the field of BPM.

The different activities in the field of BPM are often structured in a so-called BPM lifecycle. Several authors (e.g., Hallerbach et al., 2008b; Netjes et al., 2006; van der Aalst, 2004; van der Aalst et al., 2012; Weske, 2012; Wetzstein et al., 2007; zur Muehlen et al., 2006) have defined such BPM lifecycles (de Morais et al., 2014), which slightly vary in the number of phases, the names of the phases and the assignment of activities to these phases. For reasons of brevity, the dissertation at hand will not discuss the different BPM lifecycle definitions in detail but relies on the BPM lifecycle definition proposed by Wetzstein et al. (2007) (cf. Figure 1.1). This particular BPM lifecycle definition consists of four phases namely (1) process modeling, (2) process implementation, (3) process execution and (4) process analysis.

Considering the example, in the *process modeling* phase (1) a process model covering the supply chain, the assembly as well as the delivery is constructed. System analysts and process modelers analyze which parts need to be ordered from suppliers at which point during the conduction of the process, for example. When constructing as-is process models, Process Mining is supporting modelers by deriving process models from event logs (cf., e.g., van der Aalst, 2011; van der Aalst, 2015; van der Aalst et al., 2012; van Dongen et al., 2009; van Dongen et al., 2005; Weijters et al., 2006). When talking about to-be process models, Automated Planning of Process Models (cf. Heinrich et al., 2008; Heinrich et al., 2012; Henneberger et al., 2008) aims at supporting modelers by constructing process models (semi-)automatically. It combines semantic annotations as envisioned in the research area Semantic Business Process Management (SBPM) (Betz et al., 2006; Brockmans et al., 2006; Hepp et al., 2007; Hepp et al., 2005; Thomas et al., 2009) and existing approaches from AI (abbr.: artificial intelligence) planning (cf., e.g., Bertoli et al., 2001; Bertoli et al., 2006; Hoffmann et al., 2005).

In the (2) *process implementation* phase, the process model is converted (i.e., enriched and transformed) into a process that could actually be conducted. Comparable to the aforementioned process modeling notations, these executable processes are usually represented in terms of standardized machine-readable data formats like BPEL (Business Process Execution Language). (Web-)services are assigned to the actions of the process model so that the process could be conducted in a (semi-)automated manner (cf., e.g., Agarwal et al., 2005; Bashari et al., 2018; Bertoli et al., 2010; Fujii et al., 2009; Heinrich et al., 2015a; Lewerenz, 2015; Meyer et al., 2006; Wang et al., 2014; Weber, 2007). For instance, in the case of the car manufacturer, the suppliers may offer web-services that enable the automobile manufacturer to order the required cables. Hence, the corresponding process model is enriched in this phase and the web-services of the suppliers are assigned to the action "Order cables". In recent years, several approaches have evolved to support modelers and business analysts in this phase by means of algorithms and automation. Approaches from the field of (web-)service selection (cf., e.g., Ding et al., 2015; Heinrich et al., 2015a; Khan et al., 2010; Lewerenz, 2015) allow to automatically select appropriate (web-)services based on non-functional criteria, for example. Within the aforementioned example, the suppliers for cables differ in price, speed of delivery and capacity. Based on such non-functional criteria, an appropriate supplier can be selected automatically.

The *process execution* phase (3) marks the phase in which the process is actually being conducted. Hence, when a customer orders a car, for instance via the webpage of the automobile manufacturer, the according process is instantiated. Thereby, the actual order for the cables required for precisely the particular car which was ordered is automatically dispatched from the automobile manufacturer to the aforementioned or-

dering web-service of one of the suppliers. The automation of executing processes is of particular interest for practitioners in literally any business sector nowadays. A distinction is made between an "inside-out" and an "outside-in" approach. When following an inside-out approach, information systems are extended by means of (web-)services. This enables, for instance, the automated selection of services, based on quality of service criteria like execution costs (cf., e.g., Ding et al., 2015; Khan et al., 2010; Wang et al., 2014). In contrast to this, Robotic Process Automation "aims to replace people by automation done in an "outside-in" manner" (van der Aalst et al., 2018) by relying on already existing user interfaces of information systems to automatically interact with them.

The (4) *process analysis* phase finally comprises activities to monitor and analyze the process instances during their conduction and to identify possibilities for improvement. Thus, deviations from the underlying process model (e.g., due to external influences) are analyzed, for example, or the overall performance of the conduction of processes is monitored as well as analyzed and possibilities for improvement are derived. The automobile manufacturer may identify that ordered cables have been delivered later than they were supposed to in several process instances and hence, the final assembly of the according cars had to be postponed, for example. Therefore, in this phase for instance approaches are proposed that aim for automated error handling procedures to resolve process instances that are interrupted due to, for example external events (cf., e.g., Linden et al., 2014; Marrella et al., 2011a; Marrella et al., 2011b; Marrella et al., 2012; Tax et al., 2017; van Beest et al., 2014). Other works aim at addressing deviation from processes automatically (cf., e.g., Reichert et al., 1997; Reichert et al., 1998; Rinderle et al., 2004) or at adapting process models due to discrepancies which occurred during the conduction in an automated manner (cf., e.g., Garrido et al., 2010; Gerevini et al., 2000; Gerevini et al., 2012; Marrella et al., 2017; Nunes et al., 2018; Scala et al., 2015; van der Krogt et al., 2002; van der Krogt et al., 2005).

With processes becoming larger and more complex, the activities comprised by the BPM lifecycle become more complex and thus time-consuming, too (Škrinjar et al., 2010). Each phase comprises approaches, which are used during the actual conduction of process (i.e., online or at run time) and approaches that are used, when the process at hand is not currently conducted (i.e., offline or at design time). Approaches used at run time are typically more time sensitive. Approaches used at design time, prior to the actual conduction, are typically used to prepare the conduction by means of time-consuming tasks. As mentioned beforehand, companies are required to be efficient and agile (Hepp et al., 2005) to face the challenges of the modern, competitive business world. This is envisioned by reducing the required time to market as well as the "setup costs" through automation (Hepp et al., 2005). In addition, the flexibility

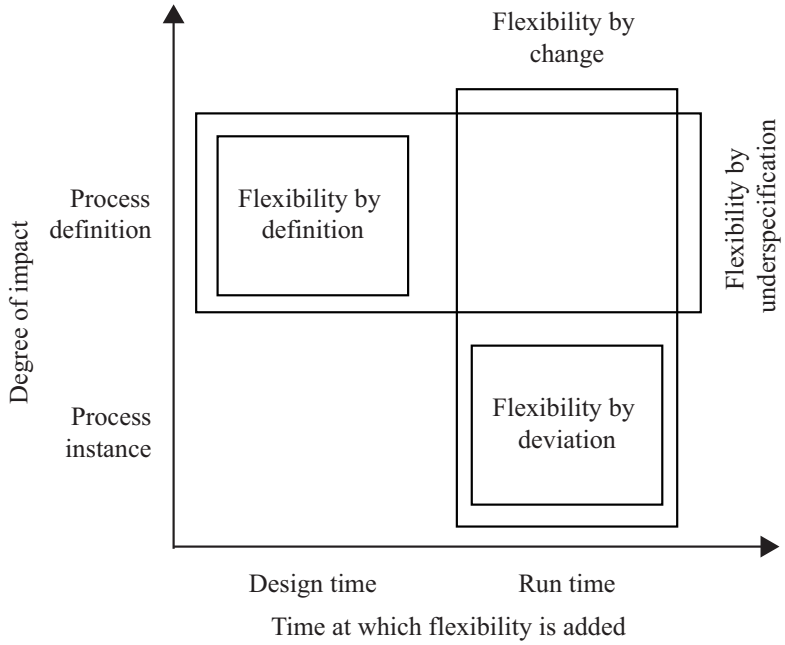

Figure 1.2: Taxonomy of Process Flexibility as presented by van der Aalst et al. (2013)

of processes is to be increased. This aspect is discussed in the following Section before dealing with the particular focus of the dissertation at hand.

1.3 Business Process Flexibility

Besides the fact that BPM could be time-consuming and hence potentially be more an impediment for agility than an enabler, processes are influenced by plenty of different factors. Due to these factors process models must necessarily become flexible to allow organizations being agile. Schonenberg et al. (2008) present a "taxonomy of process flexibility" (van der Aalst, 2013, p. 25) that "identifies four main flexibility types: flexibility by definition, flexibility by deviation, flexibility by underspecification, and flexibility by change" (van der Aalst, 2013, p. 25; cf. Figure 1.2).

These types distinguish flexibility by means of the time at which flexibility is added as well as by the degree of impact. First to mention, the time at which flexibility is added is twofold. Flexibility could be added during "design time" (i.e., the construction of a process) or during "run time" (i.e., during the conduction of a process). Secondly, the degree of impact is twofold as well. Added flexibility could impact the process definition (e.g., a process model) or the process instance, which means that the process definition stays the same but a particular, conducted process instance is altered.

Flexibility by definition describes the ability to explicitly represent different, alternative execution paths within a process model. The most common way to add flexibility by definition is to rely on well-known workflow patterns (also referred to as control flow patterns or control flow structures; Russell et al., 2006a; Russell et al., 2016; van der Aalst et al., 2003). For instance, the control flow pattern exclusive choice allows to define different branches and according conditions that have to be met in order to conduct these branches. This enables selecting an according branch at run time, based on a valuation of the current situation with regard to the conditions that hold for the branch. To give an example: In the aforementioned example of an automotive manufacturing process, particular parts such as seat heating need to be installed, only if a customer orders them. Increased flexibility by definition increases the amount of different situations that may be taken into account in one process definition.

Flexibility by deviation describes a contrary aspect. It describes the fact that entities, participating in the conduction of a process may deviate from a given process description (e.g., an underlying process model for the currently conducted process) in order to become more flexible to a current situation. A common example of flexibility by deviation is exception handling at run time. Even though there exist so-called exception handling patterns (Lerner et al., 2010; Russell et al., 2006b), exception handling is oftentimes a problem strived at run time (cf., e.g., Wang et al., 2017). Declarative process modeling is an alternative to classic, imperative process modeling and aims at increasing the flexibility by deviation as well. One of its core concepts is to "[focus] on what can be done to achieve a business goal" instead of determining "what should be done during a workflow process" (cf., e.g., van der Aalst et al., 2005). Van der Aalst et al. (2005) rely on processes in the field of hospitals, where it is nearly impossible to take each and every scenario into account at design time. A particular issue of flexibility by deviation, however, is that through deviation at run time, control may be lost for process owners. Hence, possibilities to decrease deviation by already considering them at design time (and hence as part of flexibility by definition) are discussed (cf., e.g., Bauer, 2019).

Flexibility by underspecification describes the possibility to conduct processes that are not sufficiently described by the underlying process descriptions, comprising placeholders. Van der Aalst (2013) describes two potential approaches to implement this type of flexibility: Late binding describes the implementation of a placeholder by selecting from a set of available process fragments and late modeling describes the construction of a new process fragment in order to complete a given placeholder. Képes et al. (2016), for instance, rely on process fragments in order to transform "situation-independent workflow models" into "situation-aware workflow models that cope with dynamic contextual situations". Weber et al. (2008a) propose so-called change patterns and process

fragments to increase the run time flexibility of PAIS. La Rosa et al. (2017) discuss different approaches to depict variability (flexibility with respect to different situations) in so-called configurative process models. They rely on "consolidated model[s] of process variants" and transformations that enable the customization of such configurative process models.

Flexibility by change describes the issue of altering a process definition at run time so that "one or all of the currently [conducted] process instances are migrated to a new process definition" (van der Aalst, 2013, p. 25). Van der Aalst (2013) distinguishes between momentary changes and evolutionary changes in this context. Momentary changes affect the conduction of selected process instances whereas evolutionary changes, in contrast, are changes that potentially affect all future process instances. While PAIS often cope with momentary changes by means of deviation and/or process fragments (cf., e.g., Képes et al., 2016; Weber et al., 2008a), evolutionary changes (also referred to as concept drifts) are hard to handle. There exist approaches to detect evolutionary changes from event logs (cf., e.g., Seeliger et al., 2017; Zheng et al., 2017), however a large proportion of process redesign and reengineering in the run-up of the actual demand is still a manual task.

1.4 Focus of the Dissertation

After presenting different approaches from the field of Business Process Automation along the BPM lifecycle and different types of flexibility, the focus of the dissertation, comprising two focal topics FT1 and FT2, will be introduced in this Section.

On the one hand, this work aims at contributing to decreasing the manual efforts during the process modeling phase. As already mentioned in Section 1.2, algorithms are already used to decrease the required manual efforts of modelers by automatically constructing process models during this phase. Existing works in the research strand Process Mining aim at deriving process models from event logs, for instance. Thereby, the as-is situation (how processes are actually conducted in a company) could automatically be transformed into process models. However, the first focal topic FT1 of this dissertation aims at contributing to the construction of to-be process models. The dissertation thus contributes to the research field Automated Planning of Process Mod-

els[1] and hence focusses on FT1 decreasing the required manual efforts for constructing to-be process models.[2]

On the other hand, algorithms are used to increase the flexibility of processes. "BPM systems must increase their level of automation to provide the reactivity and flexibility necessary for process management" (Marrella, 2019, p. 79). Following this objective, this dissertation aims at increasing the flexibility by definition (cf. Section 1.3). This is of particular relevance in highly regulated industries, where adherence to processes is strictly necessary as well as in the context of automated process execution in well known PAIS. Increasing flexibility by deviation brings major shortcomings as declarative process modeling does not allow to depict all possible execution alternatives in the according process model, for example. In contrast to this, it seems promising to extend the flexibility by definition of imperative process models. This enables to increase flexibility while maintaining the "inside-to-outside" (Pesic, 2008) approach and hence specifying all possible execution alternatives explicitly in the according model. As Automated Planning of Process Models potentially decreases manual efforts already and hence the required time for constructing and adapting process models is reduced, companies are able to react faster to changing demands and increase their flexibility in general. However, in addition to that, different additional factors such as environmental influences, multiple contributing actors and changing customer demands require processes to be flexible in particular. To extend the existing body of knowledge, the second focal topic FT2 of this dissertation aims at considering a selection of these factors, which, until now, have typically been considered by means of other types of flexibility, by approaches from the field Automated Planning of Process Models. To do so, FT2 aims at presenting approaches from this field that increase the flexibility by definition of imperative to-be process models by considering selected factors that require processes to be flexible.

The following Section will briefly sketch the potential of Automated Planning of Process Models to decrease manual efforts for constructing to-be process models before in Section 1.4.2 selected factors, which this dissertation addresses and which require flexibility of processes, will be introduced.

1.4.1 Decreasing Manual Efforts for constructing to-be Process Models

Automated Planning of Process Models, as already mentioned, aims at leveraging a higher degree of automation in BPM by combining semantic annotations as envisioned

[1] The abbreviation Automated Planning will be used synonymously in the remainder of the dissertation.

[2] In the remainder of this dissertation process models are structured and denoted in terms of UML Activity Diagrams. Similar or identical notations also exist in other standards such as EPC or BPMN. The application of the presented approaches is not limited to one of these process modeling notations. See the remainder for clarification.

in SBPM (Betz et al., 2006; Brockmans et al., 2006; Hepp et al., 2007; Hepp et al., 2005; Thomas et al., 2009) and existing approaches from AI planning (cf., e.g., Bertoli et al., 2001; Bertoli et al., 2006; Hoffmann et al., 2005). Thereby, process modelers are supported as to-be process models could be constructed (semi-)automatically and hence "new levels of automation" can be enabled (Marrella, 2019). In particular, Automated Planning of Process Models can be understood as a planning problem (cf. Ghallab et al., 2004; Ghallab et al., 2016) aiming at arranging modeling elements (i.e., actions and control flow patterns) in a feasible order, based on a given initial state, a set of (semantically) annotated, available actions and sets of conditions for goal states. Prior works in this field have already created a good basis, which is now to be expanded with respect to FT1.

These prior works have strongly focused on constructing feasible sequences of actions from the initial state to states fulfilling the conditions for goal states (cf., e.g., Heinrich et al., 2008; Heinrich et al., 2012; Henneberger et al., 2008). However, besides automatically constructing feasible sequences of actions, control flow patterns are essential components of process models that need to be automatically constructed as well. Here as well, prior works that present algorithms to construct selected control flow patterns such as the exclusive choice (cf., Heinrich et al., 2009; Heinrich et al., 2015b) or parallelizations (including according synchronizations; cf., Heinrich et al., 2019) already exist. However, typical process models do comprise more than these control flow patterns that may already be constructed automatically. For instance, the set of basic control flow patterns (The Workflow Management Coalition Specification, 1999) comprises the exclusive choice, parallelizations as well as the simple merge (also referred to as OR-join). It is therefore reasonable to strive to achieve the automated construction of all these basic control flow structures.

Besides that, even though previous authors state that manual efforts could be decreased by means of Automated Planning of Process Models (cf., e.g., Heinrich et al., 2008; Heinrich et al., 2009; Heinrich et al., 2012), it is still to be evaluated whether modelers are indeed more efficient when relying on Automated Planning of Process Models. Even though a higher degree of automation is envisioned, for instance, the efforts required for the (initial) annotation of actions are expected to be higher, compared to common process modeling approaches (cf., e.g., Heinrich et al., 2015b; Krause et al., 2013).

This dissertation follows previous works in this research strand and aims at extending the existing body of knowledge. The particular contribution lays in extending the capabilities in the field of automatically constructing basic control flow patterns as well as particular research on how Automated Planning of Process Models influences the efficiency of process modelers with respect to FT1.

1.4.2 Increasing the Flexibility of to-be Processes

As discussed in Section 1.3, processes are increasingly influenced by different factors that lead to the necessity of these processes to be flexible. In the following, three selected influencing factors (IF1-IF3) will be discussed, which lay the foundation for the second focal topic FT2 of this dissertation. First, new circumstances to which processes and process models need to be adapted, strongly influence the processes of companies. For instance, the process of assembling a car nowadays is changing for most car manufacturers, as e-mobility arises. Instead of assembling and mounting the combustion engine, the fuel tank, and the gearbox for fuel powered cars, for electric cars a charger, a battery, and an electric motor combined with a DC controller has to be assembled and mounted into the body of the car. Additionally, due to the emissions scandal a few years ago, car manufacturers are legally required to conduct additional emission tests (cf. Milionis et al., 2019) today. Harmon (2019) differentiates between three different levels of concern with regard to business process change. At first, he states that "organizations normally undertake a variety of specific projects to create, redesign or improve specific business processes" and refers to these projects as *process level concerns*. A typical example for this type of concern would be a business process reengineering initiative to streamline the customer service process of a company such as the aforementioned car manufacturer. Additionally, *implementation level concerns* such as the acquisition of a new ERP software, which are typically driven by IT, are conducted. This may influence the underlying processes heavily. As superordinate *enterprise level concerns*, Harmon (2019) subsumes projects and initiatives that span across whole organizations such as a wide ranging supply chain process redesign. In particular with respect to process and enterprise level concerns, the underlying processes are changed durably. That means, that changes that become necessary at a particular point in time will most likely not become superfluous any time soon. As a result, processes and their corresponding process models become infeasible due to such necessary changes, which are called *upcoming needs for change* in the remainder, and need to be adapted. Hence, *upcoming needs for change* IF1 influence processes and the corresponding process models and require them to be flexible nowadays.

However, modern process models are required to be not only adaptive to IF1 upcoming needs for change but also to different *environmental* (i.e., not considered as process-internal; also referred to as exogenous in the remainder) influences. In contrast to durable changes to processes due to IF1 upcoming needs for change, such environmental influences are often (generally speaking) shorter-term and not durable. Rosemann et al. (2008), for instance, state that "organizations often face continuous and unprecedented changes in their respective business environments". A very common yet im-

pressive and clarifying example for such environmental influences is changing weather. Even car-manufacturers got hit by environmental influences like back in 2010 when Icelandic volcano Eyjafjallajökull erupted huge amounts of ash. BMW as well as Nissan had to stop production due to shortages in the supply chain[3]. Another fairly recent example is the coronavirus crisis in 2020. Car manufacturers in Germany are using their 3D printers to support the medical industry in the production of ventilators[4]. This requires temporary additional tests to be integrated into the manufacturing processes to meet the high hygiene and safety standards of the medical industry. On the other hand, local dealers have to close their stores and even central fairs such as the Geneva Motor Show get canceled[5]. This requires car manufacturers to adapt their processes (especially in marketing and sales) quickly. In case of the canceled car show, the conduction of processes for the preparation of the fair has even very likely already started. Hence, the processes had to be adapted to changing context during their conduction. Particularly, processes have to be designed so that they are suitable for different momentary situations. Literature denotes this fact as context-awareness (Gottschalk et al., 2010; Gottschalk et al., 2007; La Rosa et al., 2011a; Reichert et al., 2012; Swenson, 2010; van der Aalst et al., 2006). Context-awareness in its most basic form could be considered by means of the control flow of process models. In particular, the control flow pattern exclusive choice enables to denote so-called branches that are conducted when a particular condition holds (Heinrich et al., 2009; Heinrich et al., 2015b; Russell et al., 2006a; Russell et al., 2016). Thus, the exclusive choice allows to denote that a particular branch has to be conducted, when it is sunny and an alternative branch must be conducted, when it is raining, for example. However, the fact that context may change during the conduction of a process and hence is non-static (cf. Dobson et al., 2006) is still an unsolved issue. This second factor, *exogenous non-static context* IF2, will therefore be taken into consideration as it heavily influences processes and requires them to be flexible.

Besides this, processes are not only carried out by singular actors (e.g., persons, departments or companies) but by multiple participating actors at once. For instance, the process of assembling a car is carried out by a plethora of different actors. First to mention, multiple people are contributing to the process alongside the assembly line. There may be a particular person, responsible for assuring the quality of the car and others

[3] https://www.theguardian.com/business/blog/2010/apr/20/nissan-suspends-car-production-volcano-ash-cloud

[4] https://www.nytimes.com/reuters/2020/03/23/world/europe/23reuters-health-coronavirus-germany-autos.html

[5] https://edition.cnn.com/2020/03/09/cars/coronavirus-geneva-international-motor-show-online/index.html

in final assembly[6]. In addition, not only one company is involved in the entire process. Mercedes Benz is relying on 25 module suppliers for their "Smart", for example (cf. Doran, 2004). Each module supplier is responsible for a particular production module such as the "driver's cockpit containing airbags, heating and air-conditioning systems, the instrument cluster, the steering column and the wiring harness" (Doran, 2004, p. 102f). However, with different actors participating in a process, those actors might follow individual goals (e.g., increasing their individual revenue) as well as shared goals (e.g., delivering a car to an end-customer). Those individual and shared goals may be conflicting or in accordance with each other. They potentially require that participating actors need to cooperate during the process to reach their goals. Hence, such influences, resulting from *multiple cooperating actors* IF3 have to be considered in modern processes as well.

In order to contribute to leveraging challenges resulting from different factors, this dissertation aims at providing approaches from the field of Automated Planning to make processes more adaptive to the aforementioned factors IF1-IF3 and thus increase their flexibility FT2 while decreasing the required manual efforts to construct the corresponding process models FT1. Hence, in the following Section, the research questions (RQ) of this dissertation will be summarized.

1.5 Research Questions

In order to address both focal topics FT1 decreasing the manual effort during the process modeling phase by means of Automated Planning of Process Models and FT2 increasing the flexibility by definition of to-be process models this dissertation aims at answering five research questions. Three of them (RQ1.1 to RQ1.3) cope with the issue of creating (adaptive) process models from scratch with respect to both focal topics FT1 and FT2. One (RQ2) is coping with the adaptation of already existing process models and hence aims at contributing to focal topic FT2 (striving IF1 in particular) and one (RQ3) is aiming at an evaluation of the Automated Planning approach with respect to FT1.

As already mentioned, there previous works on the Automated Planning of Process Models have already laid a good foundation by presenting approaches that identifying possible sequences of actions (cf., e.g., Bertoli et al., 2006; Heinrich et al., 2008; Heinrich et al., 2012). However besides that, so-called control flow patterns (Russell et al., 2006a; Russell et al., 2016) need to be constructed automatically as well, to construct complete process models. For instance, Heinrich et al. (2019) construct the control flow patterns

[6] https://www.autonews.com/article/20180123/OEM01/180129910/why-assembly-plants-need-people-more-than-robots

parallel split and synchronization automatically. In reference to constructing adaptive process models (cf. FT2), there already exists work on how to construct the control flow pattern exclusive choice (Heinrich et al., 2009; Heinrich et al., 2015b). This control flow pattern distinguishes paths in a process model, based on conditions and hence enables the construction of process models that are adaptive to different situations. With respect to the aforementioned example, an exclusive choice enables the automobile manufacturer to decide whether to install heater plugs (in case a diesel car is ordered) or spark plugs (in case a petrol car is ordered). However, it is irrelevant whether a diesel or a petrol car is ordered for the interior of the car. Hence, after installing the engine, the individual paths for assembling a diesel car and a petrol car could be combined by means of the control flow pattern simple merge as from there on they are similar. The simple merge serves as a counterpart to the exclusive choice and allows to combine individual paths (adapted to a particular situation) that are similar from a particular point. In order to contribute to FT1, this dissertation aims at constructing the control flow pattern simple merge automatically. By aiming at constructing minimal process models (i.e., process models that do not comprise more elements than necessary), this contributes to FT2 as well as adaptive process models comprising exclusive choices (cf. Heinrich et al., 2009; Heinrich et al., 2015b; Russell et al., 2016; van der Aalst et al., 2003) are typically not minimal. Hence, RQ1.1 faces the so far unsolved issue of automatically constructing the control flow pattern simple merge:

> RQ1.1 How can the control flow pattern simple merge be constructed in an automated manner?

However, processes not only need to be adaptive to particular situations but also to changing process exogenous influences. Process exogenous non-static context IF2 (cf. Section 1.4.2; Dobson et al., 2006) makes it necessary to consider the fact that a process could be conducted in different contexts (environmental situations) and particularly that these contexts may change while a process is conducted. So far, the fact that processes are exposed to IF2 exogenous non-static context has not been considered in particular within the research field of Automated Planning of Process Models. Existing approaches consider a process as a "closed system" so that external influences can not be depicted by currently existing formal foundations. Further, to date no algorithm was presented, which deals with the Automated Planning of context-aware Process Models. Hence, in the remainder of this dissertation the so far unsolved research question RQ1.2, focussing on the factor IF2 and thus on FT2, by means of Automated Planning of Process Planning, should be answered:

> RQ1.2 How can comprehensive context-aware process models be planned automatically to consider non-static context during design-time?

When talking about processes in today's distributed business world, a vast majority of these processes are conducted by more than one actor. Hence, it is not sufficient to construct process models that are capable of depicting only one contributing actor. There are existing approaches to depict processes conducted by multiple actors by incorporating so-called swimlanes *Business Process Model and Notation (BPMN): Version 2.0.2* (2013) and *OMG Unified Modeling Language TM (OMG UML): Version 2.5* (2015). However, automatically coordinating multiple contributing actors nowadays is mainly realized by decentralized approaches. Such approaches split the overall problem into sub-problems for each actor, enabling a distributed system of individual algorithms to construct individual processes for each actor. However, as actors might need to cooperate in order to reach their individual or common goals, they need to be coordinated by a central mechanism. This problem, striving IF3, may be supported by an approach incorporating Automated Planning of Process Models. Hence, this dissertation aims at answering the following research question with respect to FT2:

> RQ1.3 How can feasible process models comprising multiple different actors be constructed by means of an Automated Planning approach?

Today's business world is not only distributed but also competitive. This requires companies to be agile and thus to adapt their processes efficiently and in a timely manner (cf., e.g., Harmon, 2019) to IF1 upcoming needs for change in advance. Even though, there are existing works from the research strand of Automated Planning of Process Models (e.g., Eisenbarth, 2013; Eisenbarth et al., 2011; Lautenbacher et al., 2009) that strive related issues, none of these guarantees that the resulting process models are complete. Though approaches in other research strands – such as Process Mining – to detect deviations and therefore evolutionary change in as-is processes exist (cf., e.g., Seeliger et al., 2017; Zheng et al., 2017), no particular approach to automatically adapt to-be processes to IF1 upcoming needs for change exists. Hence, it is promising to support modelers in adapting existing process modelers to IF1 upcoming needs for change in order to increase the flexibility of process models (cf. FT2) and to increase the efficiency of process reengineering and redesign initiatives with respect to FT1. This dissertation therefore aims at answering the following research question:

> RQ2 How can process models be adapted to needs for change in advance in an automated manner?

In conclusion, Automated Planning of Process Models has so far not been evaluated with respect to the efficiency of process modelers in detail. Even though first works in this field exist (Krause et al., 2013, cf., e.g.,), no detailed research has been carried

out to evaluate whether process modelers are more or less efficient when incorporating Automated Planning of Process Models in their work routine. When talking about the efficiency of process modelers, not only the required time for finishing a modeling task but also the quality of constructed process models should be taken into account with respect to the value of BPM, one of the aforementioned major challenges. Further, Automated Planning of Process Models influences the established procedure of how process models are constructed. In particular, process modelers need to put higher efforts in formally specifying the required input for the algorithms of the Automated Planning approach but on the other hand do not need to order actions and control flow patterns appropriately and hence may save time in this phase of the overall modeling task. Hence, it has to be analyzed whether, for instance, it is appropriate to incorporate Automated Planning of Process Models in rather simple and small process modeling tasks. The last research question of this dissertation therefore aims at this issue with respect to FT1:

> RQ3 According to which criteria and in which particular situations is Automated Planning of Process Models beneficial in practice?

1.6 Research Methodology and Structure of the Dissertation

In this Section the research methodology applied for addressing these research questions is briefly sketched and the structure of the dissertation is presented.

Bertrand et al. (2002) state that "quantitative model based research can be classified as a rational knowledge generation approach (see Meredith et al., 1989)" (p. 249). They further distinguish between two distinct classes of model-based OM research. The first class, which they denote as axiomatic in line with the terminology of Meredith et al. (1989), strives for obtaining "solutions within [a] defined model and [making] sure that these solutions provide insights into the structure of the problem as defined within the model" (Bertrand et al., 2002, p. 249). Hence, in place of focusing on observing the real world and "creating a model that adequately describes the causal relationships that may exist in reality" (Bertrand et al., 2002, p. 250; i.e., descriptive empirical research), axiomatic normative research aims at producing "knowledge about how to manipulate certain variables in [a particular] model, assuming desired behavior of other variables in the model" (Bertrand et al., 2002, p. 249).

Meredith et al. (1989) present a generic framework for classifying research paradigms based on the framework of Mitroff et al. (1982). Their framework relies on two key dimensions that shape the basis for research activity. One dimension, called "rational/existential dimension" is divided into four generic perspectives that structure research by

different degrees of formalism. "These four perspectives, in order of degree of formal structures are axiomatic, logical positivist/empiricist, interpretive, and critical theory" (Meredith et al., 1989, p. 305).

Paper 1 strives the construction of simple merges (RQ1.1) and primarily follows an axiomatic approach. An existing, formal foundation (model) is extended so that the construction of simple merges becomes possible. Further, an algorithm to automatically construct the control flow pattern simple merge based on the according formal foundation is presented. Further, to evaluate the presented approach, the algorithm is prototypically implemented (logical positivist/empiricist). In addition to that, the prototypical implementation was applied to different real-world cases (interpretive).

Paper 2 addresses the Automated Planning of context-aware Process Models (RQ1.2). Here, an axiomatic approach is conducted, as well. An existing formal foundation is extended to enable the representation of exogenous non-static context. This representation is formalized and denoted in terms of mathematical definitions. Additionally, an algorithm that enables the automated construction of context-aware process models is designed and prototypically implemented (logical positivist/empiricist). To evaluate the feasibility of the formal foundation as well as the algorithm, the approach is applied to three real-world scenarios (interpretive).

Paper 3 aims at coordinating multiple contributing actors conducting a process by means of an Automated Planning approach (RQ1.3). For this, the existing formal foundation is extended to represent so-called partnerships. Further, an algorithm is presented that coordinates actors to build and disband partnerships appropriately. In addition to this axiomatic approach, a logical positivist/empiricist approach was conducted by prototypically implementing the approach as well as an interpretive approach by applying the algorithm to several real-world scenarios.

Paper 4 presents an approach to adapt existing process models to upcoming needs for change (RQ2). This means that the needs for change have not yet been implemented and the adapted process models have so far not yet been realized. In the paper, possible changes are identified and addressed. In particular, a comprehensive approach that is capable of adapting existing process models to all possible changes to a given process graph is proposed. The evaluation of the approach, comprises mathematical proofs, an application in a real-world situation as well as a simulation experiment to benchmark its runtime against planning process models from scratch.

Paper 5 strives the evaluation of the presented approach for the Automated Planning of Process Models with respect to the task performance of process modelers (RQ3). Therefore, a laboratory experiment is conducted. This laboratory experiment aims at answering the question whether (resp. in which particular situations) using the pre-

Paper	RQ	Title	Authors	Status	Outlet	Methodology
1	RQ1.1	Automated Planning of Process Models: The Construction of Simple Merges	Heinrich, Bernd Schön, Dominik	published	Proceedings of the 24th European Conference on Information Systems (ECIS 2016), Istanbul, Turkey, June 12-15, 2016	extension of model, algorithm and mathematical proofs (axiomatic); prototypical implementation (log. positivist/empiricist); application to real-world cases (interpretive)
2	RQ1.2	Automated Planning of context-aware Process Models	Heinrich, Bernd Schön, Dominik	published	Proceedings of the 23rd European Conference on Information Systems (ECIS 2015), Münster, Germany, May 26-29, 2015	extension of model, algorithm and mathematical proofs (axiomatic); prototypical implementation (log. positivist/empiricist); application to real-world cases (interpretive)
3	RQ1.3	The Cooperation of Multiple Actors within Process Models: An Automated Planning Approach	Heinrich, Bernd Schiller, Alexander Schön, Dominik	published	Journal of Decision Systems (Volume 27, Issue 4, 2019)	extension of model, algorithm and mathematical proofs (axiomatic); prototypical implementation (log. positivist/empiricist); application to real-world cases (interpretive)
4	RQ2	Adapting Process Models via an Automated Planning Approach	Heinrich, Bernd Schiller, Alexander Schön, Dominik Szubartowicz, Michael	in revision	Journal of Decision Systems	extension of model, algorithm and mathematical proofs (axiomatic); prototypical implementation and simulation experiment (log. positivist/empiricist); application to real-world cases (interpretive)
5	RQ3	The Influence of Automated Planning on the Task Performance of Process Modelers	Schön, Dominik	published	Proceedings of the 40th International Conference on Information Systems (ICIS 2019), Munich, Germany, December 15-18, 2019	laboratory experiment (logical positivist/empiricist)

Table 1.1: Overview of all Papers comprised in the Dissertation

sented approach is beneficial in contrast to using a commonly known modeling tool. To sum up, this paper follows a logical positivist/empiricist approach.

In the remainder of this dissertation the five papers are presented (cf. Table 1.1). Here, Chapter 2 copes with the issue of creating (adaptive) process models from scratch with respect to both focal topics FT1 and FT2 and covers RQ1.1 to RQ1.3. Chapter 3 is coping with the adaptation of already existing process models and hence covers RQ2. Chapter 4 covers RQ3, the evaluation of the Automated Planning approach. Finally, Chapter 5 concludes the dissertation with a discussion of major findings and an outlook on further research.

1.7 References

Agarwal, V., Chafle, G., Dasgupta, K., Karnik, N., Kumar, A., Mittal, S., and Srivastava, B. (2005). "Synthy: A system for end to end composition of web services." In: *Journal of Web Semantics* 3.4. World Wide Web Conference 2005——Semantic Web Track, pp. 311–339. ISSN: 1570-8268. DOI: 10.1016/j.websem.2005.09.002.

Aguilar-Savén, R. S. (2004). "Business process modelling: Review and framework." In: *International Journal of Production Economics* 90.2, pp. 129–149. ISSN: 09255273. DOI: 10.1016/S0925-5273(03)00102-6.

Armistead, C. and Machin, S. (Jan. 1997). "Implications of business process management for operations management." In: *International Journal of Operations & Production Management* 17.9, pp. 886–898. DOI: 10.1108/01443579710171217.

Bach, M. P., Vukšić, V. B., Vugec, D. S., and Stjepić, A.-M. (Sept. 2019). "BPM and BI in SMEs: The role of BPM/BI alignment in organizational performance." In: *International Journal of Engineering Business Management* 11. DOI: 10.1177/1847979019874182.

Bandara, W., Alibabaei, A., and Aghdasi, M. (Sept. 2009). "Means of achieving Business Process Management success factors." In: *Proceedings of the 4th Mediterranean Conference on Information Systems*. Ed. by P. Ein-Dor, A. Poulymenakou, and M. Amami. Athens University of Economics and Business, Athens, Greece: Department of Management Science & Technology, Athens University of Economics and Business, pp. 25–27. URL: http://eprints.qut.edu.au/30074/.

Bandara, W., Indulska, M., Chong, S., and Sadiq, S. (2007). "Major issues in business process management: an expert perspective." In: *Proceedings of the 15th European Conference on Information Systems (ECIS)*, p. 89.

Bashari, M., Bagheri, E., and Du, W. (Dec. 2018). "Automated composition and optimization of services for variability-intensive domains." In: *Journal of Systems and Software* 146, pp. 356–376. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.

2018.07.039. URL: http://www.sciencedirect.com/science/article/pii/S0164121218301481.

Bauer, T. (2019). "Business Processes with Pre-modelled Flexibility: Examples and Requirements for Various Process Aspects." In: *American Journal of Management* 19.4.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., and Jeffries, R. (2001). *The Agile Manifesto*. URL: http://www.agilemanifesto.org.

Becker, J., Thome, I., Weiß, B., and Winkelmann, A. (July 2010a). "Constructing a Semantic Business Process Modelling Language for the Banking Sector: An Evolutionary Dyadic Design Science Approach." In: *Enterprise Modelling and Information Systems Architectures* 5.1, pp. 4–25. DOI: 10.18417/emisa.5.1.1. URL: https://www.emisa-journal.org/emisa/article/view/64.

Becker, J., Weiß, B., and Winkelmann, A. (2010b). "Utility vs. Efforts of Business Process Modeling — An Exploratory Survey in the Financial Sector." In: *Proceedings of the Multikonferenz Wirtschaftsinformatik 2010*. Ed. by M. Schumann, L. M. Kolbe, M. H. Breitner, and A. Frerichs. Göttingen: Universitätsverlag Göttingen, pp. 41–54. ISBN: 978-3-941875-31-9. URL: http://udoo.uni-muenster.de/downloads/publications/2361.pdf.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bertoli, P., Pistore, M., and Traverso, P. (2010). "Automated composition of Web services via planning in asynchronous domains." In: *Artificial Intelligence* 174.3-4, pp. 316–361. ISSN: 0004-3702. DOI: 10.1016/j.artint.2009.12.002.

Bertrand, J. W. M. and Fransoo, J. C. (2002). "Operations management research methodologies using quantitative modeling." In: *International Journal of Operations & Production Management* 22.2, pp. 241–264.

Betz, S., Klink, S., Koschmider, A., and Oberweis, A. (2006). "Automatic user support for business process modeling." In: *Proceedings of the 3rd European Semantic Web Conference (ESWC 2006)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.83.4899.

Bowers, J., Button, G., and Sharrock, W. (1995). "Workflow From Within and Without: Technology and Cooperative Work on the Print Industry Shopfloor." In: *Proceedings of*

the Fourth European Conference on Computer-Supported Cooperative Work (ECSCW '95). Ed. by H. Marmolin, Y. Sundblad, and K. Schmidt. Dordrecht: Springer Netherlands, pp. 51–66. ISBN: 978-94-011-0349-7. DOI: 10.1007/978-94-011-0349-7_4.

Branco, M. C., Xiong, Y., Czarnecki, K., Küster, J., and Völzer, H. (2014). "A case study on consistency management of business and IT process models in banking." In: *Software & Systems Modeling* 13.3, pp. 913–940. ISSN: 1619-1366. DOI: 10.1007/s10270-013-0318-8.

Braunnagel, D., Falk, T., Wehner, B., and Leist, S. (June 2016). "BPM Adoption in Small and Medium-sized Companies in Bavaria." In: *Proceedings of the 24th European Conference on Information Systems (ECIS)*. URL: https://epub.uni-regensburg.de/34064/.

Brockmans, S., Ehrig, M., Koschmider, A., Oberweis, A., and Studer, R. (2006). "Semantic Alignment Of Business Processes." In: *Proceedings of the Eighth International Conference on Enterprise Information Systems (ICEIS 2006)*, pp. 191–196.

Business Process Model and Notation (BPMN): Version 2.0.2 (2013). URL: http://www.omg.org/spec/BPMN/2.0.2/PDF (visited on 10/16/2014).

Davenport, T. H. (1993). *Process innovation: Reengineering work through information technology*. Boston, Mass.: Harvard Business School Press. ISBN: 9780875843667.

De Morais, R. M., Kazan, S., de Pádua, S. I. D., and Costa, A. L. (2014). "An analysis of BPM lifecycles: from a literature review to a framework proposal." In: *Business Process Mgmt Journal* 20.3, pp. 412–432. ISSN: 1463-7154. DOI: 10.1108/BPMJ-03-2013-0035.

Ding, Z., Sun, Y., Liu, J., Pan, M., and Liu, J. (2015). "A genetic algorithm based approach to transactional and QoS-aware service selection." In: *Enterprise Information Systems*, pp. 1–20. ISSN: 1751-7575. DOI: 10.1080/17517575.2015.1048832.

Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). "A survey of autonomic communications." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 1.2, pp. 223–259. ISSN: 1556-4665.

Doran, D. (2004). "Rethinking the supply chain: an automotive perspective." In: *Supply Chain Management: An International Journal* 9.1, pp. 102–109.

Eisenbarth, T. (2013). *Semantic Process Models: Transformation, Adaptation, Resource Consideration*. Augsburg: Universität Augsburg.

Eisenbarth, T., Lautenbacher, F., and Bauer, B. (2011). "Adaptation of Process Models – A Semantic-based Approach." In: *Journal of Research and Practice in Information Technology* 43.1, pp. 5–23.

Fujii, K. and Suda, T. (2009). "Semantics-based context-aware dynamic service composition." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 4.2, p. 12. ISSN: 1556-4665.

Garrido, A., Guzman, C., and Onaindia, E. (2010). "Anytime plan-adaptation for continuous planning." In: *PlanSIG'10*, pp. 62–69.

Gerevini, A. and Serina, I. (Apr. 2000). "Fast Plan Adaptation through Planning Graphs: Local and Systematic Search Techniques." In: *Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems*. AIPS'00. Breckenridge, CO, USA: AAAI Press, pp. 112–121. ISBN: 1577351118.

Gerevini, A. E., Saetti, A., and Serina, I. (2012). "Case-based Planning for Problems with Real-valued Fluents: Kernel Functions for Effective Plan Retrieval." In: *Frontiers in Artificial Intelligence and Applications* 242.ECAI 2012, pp. 348–353.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Ghallab, M., Nau, D. S., and Traverso, P. (2016). *Automated Planning and Acting*. New York, NY: Cambridge University Press. ISBN: 978-1107037274.

Glavan, L. M. and Vukšić, V. B. (2017). "Examining the impact of business process orientation on organizational performance: the case of Croatia." In: *Croatian Operational Research Review* 8.1, pp. 137–165.

Goldman, S. L., Nagel, R. N., and Preiss, K. (1994). *Agile Competitors and Virtual Organisations: Strategies for Enriching the Customer*. Van Nostrand Reinhold.

Gottschalk, F. and La Rosa, M. (2010). "Process Configuration." In: *Modern Business Process Automation*. Ed. by A. H. M. ter Hofstede, W. M. P. van der Aalst, M. Adams, and N. Russell. Berlin Heidelberg: Springer, pp. 459–487. ISBN: 978-3-642-03122-9. DOI: 10.1007/978-3-642-03121-2_18.

Gottschalk, F., van der Aalst, W. M. P., and Jansen-Vullers, M. H. (2007). "Configurable process models—a foundational approach." In: *Reference Modeling*. Springer, pp. 59–77. ISBN: 3790819654.

Hallerbach, A., Bauer, T., and Reichert, M. (June 2008b). "Managing Process Variants in the Process Lifecycle." In: *Proceedings of the 10th International Conference on Enterprise Information Systems*. Barcelona, Spain, pp. 154–161.

Harmon, P. (2019). *Business Process Change - A Business Process Management Guide for Managers and Process Professionals*. Fourth Edition. Morgan Kaufmann.

Harraf, A., Wanasika, I., Tate, K., and Talbott, K. (2015). "Organizational agility." In: *Journal of Applied Business Research* 31.2, p. 675.

Heinrich, B., Bewernik, M.-A., Henneberger, M., Krammer, A., and Lautenbacher, F. (2008). "SEMPA - A Semantic Business Process Management Approach for the Planning of Process Models." In: *Business & Information Systems Engineering* (*formerly Wirtschaftsinformatik*) 50.6, 445–460 (in German). URL: http://epub.uni-regensburg.de/23173/.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems* (*ICIS*). Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., Lewerenz, L., and Zimmermann, S. (Mar. 2015a). "Quality of Service-aware Service Selection: A Novel Approach Considering Potential Service Failures and Non-Deterministic Service Values." In: *INFORMS Service Science, A Journal of the Institute for Operations Research and the Management Sciences* 7.1, pp. 48–69. DOI: 10.1287/serv.2015.0093.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Heinrich, B., Krause, F., and Schiller, A. (2019). "Automated planning of process models: The construction of parallel splits and synchronizations." In: *Decision Support Systems* 125, p. 113096. ISSN: 0167-9236. DOI: 10.1016/j.dss.2019.113096. URL: http://www.sciencedirect.com/science/article/pii/S0167923619301253.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik* (*MKWI*). Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hepp, M. and Dumitri, R. (2007). "An Ontology Framework for Semantic Business Process Management." In: *Proceedings of the 8th International Conference on Business Informatics* (*WI 2007*)*: eOrganisation: Service-*, pp. 423–440.

Hepp, M., Leymann, F., Bussler, C., Domingue, J., Wahler, A., and Fensel, D. (2005). "Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management." In: *IEEE International Conference on E-Business Engineering* (*ICEBE 2005*) 0, pp. 535–540. DOI: 10.1109/ICEBE.2005.110.

Hoffmann, J. and Brafman, R. I. (2005). "Contingent Planning via Heuristic Forward Search with Implicit Belief States." In: *Proceedings of the 15th International Conference on Automated Planning and Scheduling*. ICAPS'05. AAAI Press, pp. 71–88. ISBN: 1-57735-220-3.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference on Advanced Information Systems Engineering (CAiSE)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

Indulska, M., Recker, J., Rosemann, M., and Green, P. (2009). "Business Process Modeling: Current Issues and Future Challenges." In: *Advanced Information Systems Engineering*. Ed. by P. van Eck, J. Gordijn, and R. Wieringa. Vol. 5565. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 501–514. ISBN: 978-3-642-02143-5. DOI: 10.1007/978-3-642-02144-2_39.

Jin-Hai, L., Anderson, A. R., and Harrison, R. T. (2003). "The evolution of agile manufacturing." In: *Business Process Management Journal* 9.2, pp. 170–189.

Keller, G., Nüttgens, M., and Scheer, A.-W. (1992). "Semantische Prozeßmodellierung auf der Grundlage'Ereignisgesteuerter Prozeßketten (EPK)." In: *Veröffentlichungen des Instituts für Wirtschaftsinformatik*. Vol. 89. Saarbrücken: Universität Saarbrücken.

Képes, K., Breitenbücher, U., Gómez Sáez, S., Guth, J., Leymann, F., and Wieland, M. (2016). "Situation-Aware Execution and Dynamic Adaptation of Traditional Workflow Models." In: *Service-Oriented and Cloud Computing*. Ed. by M. Aiello, E. B. Johnsen, S. Dustdar, and I. Georgievski. Cham: Springer International Publishing, pp. 69–83. ISBN: 978-3-319-44482-6. DOI: 10.1007/978-3-319-44482-6_5.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security (IJCSIS)* 7.2, pp. 147–152.

Krause, F., Bewernik, M.-A., and Fridgen, G. (2013). "Valuation of Manual and Automated Process Redesign from a Business Perspective." In: *Business Process Management Journal* 19.1.

La Rosa, M., Dumas, M., ter Hofstede, A. H. M., and Mendling, J. (2011a). "Configurable multi-perspective business process models." In: *Information Systems* 36.2, pp. 313–340. ISSN: 0306-4379.

La Rosa, M., van der Aalst, W. M. P., Dumas, M., and Milani, F. P. (Mar. 2017). "Business Process Variability Modeling: A Survey." In: *ACM Comput. Surv.* 50.1. ISSN: 0360-0300. DOI: 10.1145/3041957.

Lautenbacher, F., Eisenbarth, T., and Bauer, B. (2009). "Process model adaptation using semantic technologies." In: *2009 13th Enterprise Distributed Object Computing Conference Workshops, EDOCW*, pp. 301–309. DOI: 10.1109/EDOCW.2009.5331985.

Lerner, B. S., Christov, S., Osterweil, L. J., Bendraou, R., Kannengiesser, U., and Wise, A. (2010). "Exception handling patterns for process modeling." In: *Software Engineering, IEEE Transactions on* 36.2, pp. 162–183. ISSN: 0098-5589.

Lewerenz, L. (2015). "A Heuristic Technique for an Efficient Decision Support in Context-aware Service Selection." In: *Proceedings of the 36th International Conference on Information Systems (ICIS)*, pp. 1–20. URL: https://epub.uni-regensburg.de/32660/.

Linden, I., Derbali, M., Schwanen, G., Jacquet, J.-M., Ramdoyal, R., and Ponsard, C. (2014). "Supporting Business Process Exception Management by Dynamically Building Processes Using the BEM Framework." In: *Decision Support Systems III - Impact of Decision Support Systems for Global Environments*. Ed. by F. Dargam, J. E. Hernández, P. Zaraté, S. Liu, R. Ribeiro, B. Delibašić, and J. Papathanasiou. Vol. 184. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, pp. 67–78. ISBN: 978-3-319-11363-0. DOI: 10.1007/978-3-319-11364-7_7.

Lusk, S., Paley, S., and Spanyi, A. (June 2005). "The Evolution of Business Process Management as a Professional Discipline." In: *BPTrends*. Association of Business Process Management Professionals International.

Marrella, A. (June 2019). "Automated Planning for Business Process Management." In: *Journal on Data Semantics* 8.2, pp. 79–98. ISSN: 1861-2040. DOI: 10.1007/s13740-018-0096-0.

Marrella, A. and Mecella, M. (2011a). "Continuous Planning for Solving Business Process Adaptivity." In: *Enterprise, Business-Process and Information Systems Modeling*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, T. Halpin, S. Nurcan, J. Krogstie, P. Soffer, E. Proper, R. Schmidt, and I. Bider. Vol. 81. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 118–132. ISBN: 978-3-642-21758-6. DOI: 10.1007/978-3-642-21759-3_9.

Marrella, A., Mecella, M., and Russo, A. (2011b). "Featuring Automatic Adaptivity through Workflow Enactment and Planning." In: *Proceedings of the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Ed. by D. Georgakopoulos and J. Joshi. DOI: 10.4108/icst.collaboratecom.2011.247096.

Marrella, A., Mecella, M., and Sardina, S. (2017). "Intelligent Process Adaptation in the SmartPM System." In: *ACM Transactions on Intelligent Systems and Technology* 8.2, pp. 1–43. ISSN: 21576904. DOI: 10.1145/2948071.

Marrella, A., Russo, A., and Mecella, M. (2012). "Planlets: Automatically Recovering Dynamic Processes in YAWL." In: *On the Move to Meaningful Internet Systems: OTM 2012*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, H. Panetto, T. Dillon, S. Rinderle-Ma, P. Dadam, X. Zhou, S. Pearson, A. Ferscha, S. Bergamaschi, and I. F. Cruz. Vol. 7565. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 268–286. ISBN: 978-3-642-33605-8. DOI: 10.1007/978-3-642-33606-5_17.

Mathiesen, P., Bandara, W., Delavari, H., Harmon, P., and Brennan, K. (2011). "A comparative analysis of business analysis (BA) and business process management (BPM) capabilities." In: *Proceedings of the 19th European Conference on Information Systems (ECIS)*. Helsinki, Finland, p. 26.

Mendling, J., Reijers, H. A., and van der Aalst, W. M. P. (2010). "Seven process modeling guidelines (7PMG)." In: *Information and Software Technology* 52.2, pp. 127–136. ISSN: 0950-5849.

Mendling, J., Sánchez-González, L., García, F., and La Rosa, M. (2012a). "Thresholds for error probability measures of business process models." In: *Journal of Systems and Software* 85.5, pp. 1188–1197. DOI: 10.1016/j.jss.2012.01.017.

Meredith, J. R., Raturi, A., Amoako-Gyampah, K., and Kaplan, B. (1989). "Alternative research paradigms in operations." In: *Journal of operations management* 8.4, pp. 297–326. ISSN: 0272-6963.

Meyer, H. and Weske, M. (2006). "Automated service composition using heuristic search." In: *Business Process Management*, pp. 81–96.

Milionis, N., Jereb, S., Henderson, K., Vrabic, J., Bain, M., Dolezal, J., Roessing, E., Dos Santos, J. N. C., Simeonova, R., and Otto, J. (Feb. 2019). *The EU's response to the "dieselgate" scandal*. URL: https://www.eca.europa.eu/lists/ecadocuments/brp_vehicle_emissions/brp_vehicle_emissions_en.pdf.

Mitroff, I. I. and Mason, R. O. (1982). "Business Policy and Metaphysics: Some Philosophical Considerations." In: *Academy of Management Review* 7.3, pp. 361–371. DOI: 10.5465/amr.1982.4285320.

Netjes, M., Reijers, H. A., and van der Aalst, W. M. P. (2006). "Supporting the BPM lifecycle with FileNet." In: *Proceedings of the EMMSAD Workshop at the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06)*. Vol. 6, pp. 497–508.

Nunes, V. T., Santoro, F. M., Werner, C. M. L., and G. Ralha, C. (2018). "Real-Time Process Adaptation: A Context-Aware Replanning Approach." In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.1, pp. 99–118. ISSN: 2168-2216. DOI: 10.1109/TSMC.2016.2591538.

Nüttgens, M. and Rump, F. J. (2002). "Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK)." In: *Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen*. Ed. by J. Desel and M. Weske. LN in Informatics. GI-Edition.

Nüttgens, M. and Zimmermann, V. (1998). "Geschäftsprozeßmodellierung mit der objektorientierten Ereignisgesteuerten Prozeßkette (oEPK)." In: *Informationsmodellierung: Referenzmodelle und Werkzeuge*. Ed. by M. Maicher and H.-J. Scheruhn. Wiesbaden: Deutscher Universitätsverlag, pp. 23–35. ISBN: 978-3-663-07676-6. DOI: 10.1007/978-3-663-07676-6_2. URL: https://doi.org/10.1007/978-3-663-07676-6_2.

OMG Unified Modeling Language TM (OMG UML): Version 2.5 (2015). URL: http://www.omg.org/spec/UML/2.5 (visited on 05/04/2016).

Pesic, M. (2008). "Constraint-based workflow management systems : shifting control to users." English. Proefschrift. PhD thesis. Department of Industrial Engineering Innovation Sciences. ISBN: 978-90-386-1319-2. DOI: 10.6100/IR638413.

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

Pichler, P., Weber, B., Zugal, S., Pinggera, J., Mendling, J., and Reijers, H. (Aug. 2011). "Imperative versus Declarative Process Modeling Languages: An Empirical Investigation." In: *BPM 2011 Workshops, Part I, LNBIP* 99, pp. 383–394. DOI: 10.1007/978-3-642-28108-2_37.

Pourmirza, S., Peters, S., Dijkman, R., and Grefen, P. (Feb. 2019). "BPMS-RA: A Novel Reference Architecture for Business Process Management Systems." In: *ACM Trans. Internet Technol.* 19.1. ISSN: 1533-5399. DOI: 10.1145/3232677.

Powell, S. G., Schwaninger, M., and Trimble, C. (2001). "Measurement and control of business processes." In: *System Dynamics Review* 17.1, pp. 63–91. ISSN: 0883-7066. DOI: 10.1002/sdr.206.

Prescher, J., Di Ciccio, C., and Mendling, J. (Nov. 2014). "From Declarative Processes to Imperative Models." In: *Proceedings of the Fourth International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2014)*. Vol. 1293. Milan, Italy. DOI: 10.13140/2.1.1577.4409.

Reichert, M. and Dadam, P. (1997). "A framework for dynamic changes in workflow management systems." In: *Database and Expert Systems Applications, 1997. Proceedings., Eighth International Workshop on*, pp. 42–48.

Reichert, M. and Dadam, P. (1998). "Adeptflex—Supporting Dynamic Changes of Workflows Without Losing Control." In: *Journal of Intelligent Information Systems* 10.2, pp. 93–129. ISSN: 09259902. DOI: 10.1023/A:1008604709862.

Reichert, M. U. and Weber, B. (2012). *Enabling flexibility in process-aware information systems: challenges, methods, technologies*. Springer. ISBN: 3642304095.

Rinderle, S., Reichert, M., and Dadam, P. (2004). "Correctness criteria for dynamic changes in workflow systems—-a survey." In: *Data & Knowledge Engineering* 50.1, pp. 9–34. DOI: 10.1016/j.datak.2004.01.002.

Rosemann, M. and Bruin, T. D. (2005). "Towards a Business Process Management Maturity Model." In: *Proceedings of the Thirteenth European Conference on Information Systems (ECIS 2005)*. Ed. by F. Rajola, D. Avison, R. Winter, J. Becker, P. Ein-Dor, D. Bartmann, F. Bodendorf, C. Weinhardt, and J. Kallinikos. Verlag and the London School of Economics, pp. 1–12. URL: https://eprints.qut.edu.au/25194/.

Rosemann, M., Recker, J. C., and Flender, C. (2008). "Contextualisation of business processes." In: *International Journal of Business Process Integration and Management* 3.1, pp. 47–60. ISSN: 1741-8771.

Rosemann, M. and vom Brocke, J. (2015). "The Six Core Elements of Business Process Management." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Heidelberg Dordrecht London New York: Springer, pp. 107–122.

Russell, N., ter Hofstede, A. H. M., and Mulyar, N. (2006a). "Workflow ControlFlow Patterns: A Revised View." In: *BPM Center Report BPM-06-22*. URL: http://bpmcenter.org/reports.

Russell, N., van der Aalst, W. M. P., and Ter Hofstede, A. (2016). *Workflow patterns: The definitive guide*. Information systems. Cambridge, Massachusetts: The MIT Press. ISBN: 978-0-262-02982-7.

Russell, N., van der Aalst, W. M. P., and ter Hofstede, A. H. M. (2006b). "Exception Handling Patterns." In: *Process-Aware Information Systems. Technical report, BPM Center Report BPM-06-04, BPMcenter. org*.

Scala, E., Micalizio, R., and Torasso, P. (2015). "Robust plan execution via reconfiguration and replanning." In: *AI Communications* 28.3, pp. 479–509. ISSN: 18758452. DOI: 10.3233/AIC-140629.

Schonenberg, H., Mans, R., Russell, N., Mulyar, N., and van der Aalst, W. M. P. (2008). "Process flexibility: A survey of contemporary approaches." In: *Advances in Enterprise Engineering I*. Springer, pp. 16–30.

Seeliger, A., Nolle, T., and Mühlhäuser, M. (Mar. 2017). "Detecting Concept Drift in Processes using Graph Metrics on Process Graphs." In: *Proceedings of the 9th International Conference on Subject-Oriented Business Process Management (S-BPM ONE '17)*.

Shah, R. and Ward, P. T. (2007). "Defining and developing measures of lean production." In: *Journal of operations management* 25.4, pp. 785–805.

Škrinjar, R., Vukšić, V., and Štemberger, M. (2010). "Adoption of Business Process Orientation Practices: Slovenian and Croatian Survey." In: *Business Systems Research* 1.1-2, pp. 5–19. ISSN: 1847-9375. DOI: 10.2478/v10305-012-0022-0.

Smith, H. and Fingar, P. (2003). *Business process management: The third wave*. 1st ed. Tampa, Fla.: Meghan-Kiffer Press. ISBN: 0929652339.

Swenson, K. D. (2010). "Mastering the Unpredictable." In: *How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done, Glossary*, pp. 314–317.

Tax, N., Verenich, I., La Rosa, M., and Dumas, M. (2017). "Predictive Business Process Monitoring with LSTM Neural Networks." In: *Proceedings of the 29th International Conference on Advanced Information Systems Engineering (CAiSE)*. Ed. by E. Dubois and K. Pohl. Vol. 10253. Lecture Notes in Computer Science. Cham: Springer International

Publishing, pp. 477–492. ISBN: 978-3-319-59536-8. DOI: 10.1007/978-3-319-59536-8_30.

Taylor, F. (1911). *The Principles of Scientific Management*. Harper Brothers Publishers.

Thakur, R. and Archana, T. (Apr. 2019). "Enabling Technologies and Applications of the Internet of Things." In: *Proceedings of Recent Advances in Interdisciplinary Trends in Engineering Applications (RAITEA)*. DOI: 10.2139/ssrn.3365534.

The Workflow Management Coalition Specification (1999). *Terminology & Glossary: WFMC-TC-1011 (Issue 3.0)*.

Thomas, O. and Fellmann, M. (2009). "Semantic Process Modeling – Design and Implementation of an Ontology-based Representation of Business Processes." In: *Business & Information Systems Engineering* 1.6, pp. 438–451. ISSN: 1867-0202. DOI: 10.1007/s12599-009-0078-8.

Van der Aalst, W. M. P. (2004). "Business process management: A personal view." In: *Business Process Management Journal* 10.2, pp. 248–253. ISSN: 1463-7154. DOI: 10.1108/bpmj.2004.15710baa.001.

Van der Aalst, W. M. P. (2009). "Process-Aware Information Systems: Lessons to Be Learned from Process Mining." In: *Transactions on Petri Nets and Other Models of Concurrency II: Special Issue on Concurrency in Process-Aware Information Systems*. Ed. by K. Jensen and W. M. P. van der Aalst. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–26. ISBN: 978-3-642-00899-3. DOI: 10.1007/978-3-642-00899-3_1. URL: https://doi.org/10.1007/978-3-642-00899-3_1.

Van der Aalst, W. M. P. (2011). *Process mining: discovery, conformance and enhancement of business processes*. Springer Science & Business Media.

Van der Aalst, W. M. P. (2013). "Business process management: A comprehensive survey." In: *ISRN Software Engineering* 2013.

Van der Aalst, W. M. P. (2015). "Extracting Event Data from Databases to Unleash Process Mining." In: *BPM - Driving Innovation in a Digital World*. Ed. by J. vom Brocke and T. Schmiedel. Management for Professionals. Cham: Springer International Publishing, pp. 105–128. ISBN: 978-3-319-14429-0. DOI: 10.1007/978-3-319-14430-6_8.

Van der Aalst, W. M. P., Adriansyah, A., de Medeiros, A., Arcieri, F., Baier, T., Blickle, T., Bose, J., van den Brand, P., Brandtjen, R., and Buijs, J. (2012). "Process Mining Manifesto." In: *BPM 2011 Workshops, Part I, LNBIP 99*, pp. 169–194.

Van der Aalst, W. M. P., Bichler, M., and Heinzl, A. (2018). "Robotic Process Automation." In: *Business & Information Systems Engineering* 60.4, pp. 269–272. DOI: 10.1007/s12599-018-0542-4.

Van der Aalst, W. M. P., Dreiling, A., Gottschalk, F., Rosemann, M., and Jansen-Vullers, M. H. (2006). "Configurable process models as a basis for reference modeling." In:

Business Process Management Workshops. Ed. by C. J. Bussler and A. Haller. Springer, pp. 512–518. ISBN: 3540325956.

Van der Aalst, W. M. P., Mylopoulos, J., Rosemann, M., Shaw, M. J., Szyperski, C., La Rosa, M., and Soffer, P., eds. (2013). *Business Process Management Workshops*. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-36284-2. DOI: 10.1007/978-3-642-36285-9.

Van der Aalst, W. M. P., Pesic, M., and Schonenberg, H. (2009). "Declarative workflows: Balancing between flexibility and support." In: *Computer Science - Research and Development* 23.2, pp. 99–113.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Van der Aalst, W. M. P., Weske, M., and Grünbauer, D. (2005). "Case handling: a new paradigm for business process support." In: *Data Knowledge Engineering* 53.2, pp. 129–162. ISSN: 0169-023X. DOI: 10.1016/j.datak.2004.07.003.

Van der Krogt, R., Bos, A., and Witteveen, C. (2002). "Replanning in a Resource-Based Framework." In: *Multi-Agent Systems and Applications II*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, V. Mařík, O. Štěpánková, H. Krautwurmová, and M. Luck. Vol. 2322. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 148–158. ISBN: 978-3-540-43377-4. DOI: 10.1007/3-540-45982-0_7.

Van der Krogt, R. and de Weerdt, M. (2005). "Plan Repair as an Extension of Planning." In: *ICAPS 2005: Proceedings of the 15th International Conference on Automated Planning and Scheduling, Monterey, California, USA, 5-10 June 2005*, pp. 161–170.

Van Beest, N., Kaldeli, E., Bulanov, P., Wortmann, J. C., and Lazovik, A. (2014). "Automated runtime repair of business processes." In: *Information Systems* 39, pp. 45–79. ISSN: 0306-4379. DOI: 10.1016/j.is.2013.07.003.

Van Dongen, B. F., Alves de Medeiros, A. K., and Wen, L. (2009). "Process Mining: Overview and Outlook of Petri Net Discovery Algorithms." In: *Transactions on Petri Nets and Other Models of Concurrency II*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, K. Jensen, and W. M. P. van der Aalst. Vol. 5460. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 225–242. ISBN: 978-3-642-00898-6. DOI: 10.1007/978-3-642-00899-3_13.

Van Dongen, B. F., de Medeiros, A. K. A., Verbeek, H. M. W., Weijters, A. J. M. M., and van der Aalst, W. M. P. (2005). "The ProM Framework: A New Era in Process Mining Tool Support." In: *Applications and Theory of Petri Nets 2005*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz,

C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, G. Ciardo, and P. Darondeau. Vol. 3536. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 444–454. ISBN: 978-3-540-26301-2. DOI: 10.1007/11494744_25.

Vom Brocke, J., Schmiedel, T., Recker, J., Trkman, P., Mertens, W., and Viaene, S. (2014). "Ten principles of good business process management." In: *Business Process Management Journal* 20.4, pp. 530–548. ISSN: 1463-7154. DOI: 10.1108/BPMJ-06-2013-0074.

Wang, P., Ding, Z., Jiang, C., and Zhou, M. (2014). "Automated web service composition supporting conditional branch structures." In: *Enterprise Information Systems* 8.1, pp. 121–146. ISSN: 1751-7575. DOI: 10.1080/17517575.2011.584132.

Wang, X., Feng, Z., Huang, K., and Tan, W. (2017). "An automatic self-adaptation framework for service-based process based on exception handling." In: *Concurrency and Computation: Practice and Experience* 29.5. e3984 CPE-15-0402.R2, e3984. DOI: 10.1002/cpe.3984.

Weber, B., Reichert, M., and Rinderle-Ma, S. (2008a). "Change patterns and change support features–enhancing flexibility in process-aware information systems." In: *Data & Knowledge Engineering* 66.3, pp. 438–466.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Weijters, A., van der Aalst, W. M. P., and de Medeiros, A. A. (2006). "Process mining with the heuristics miner-algorithm." In: *Technische Universiteit Eindhoven, Tech. Rep. WP* 166.

Weske, M. (2012). *Business Process Management: Concepts, Languages, Architectures*. 2nd ed. Heidelberg Dordrecht London New York: Springer.

Wetzstein, B., Ma, Z., Filipowska, A., Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., and Cicurel, L. (2007). "Semantic Business Process Management: A Lifecycle Based Requirements Analysis." In: *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007) in conjunction with the 3rd European Semantic Web Conference (ESWC 2007)*.

Wu, K.-J., Tseng, M.-L., Chiu, A. S., and Lim, M. K. (2017). "Achieving competitive advantage through supply chain agility under uncertainty: A novel multi-criteria decision-making structure." In: *International Journal of Production Economics* 190, pp. 96–107.

Zairi, M. (1997). "Business process management: a boundaryless approach to modern competitiveness." In: *Business Process Management Journal* 3.1, pp. 64–80.

Zhang, Z. and Sharifi, H. (2000). "A methodology for achieving agility in manufacturing organisations." In: *International Journal of Operations & Production Management* 20.4, pp. 496–513.

Zheng, C., Wen, L., and Wang, J. (2017). "Detecting Process Concept Drifts from Event Logs." In: *On the Move to Meaningful Internet Systems. OTM 2017 Conferences*. Ed. by H. Panetto, C. Debruyne, W. Gaaloul, M. Papazoglou, A. Paschke, C. A. Ardagna, and R. Meersman. Cham: Springer International Publishing, pp. 524–542. ISBN: 978-3-319-69462-7.

Zur Muehlen, M. and Ho, D. T.-Y. (2006). "Risk Management in the BPM Lifecycle." In: *Business Process Management Workshops*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. J. Bussler, and A. Haller. Vol. 3812. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 454–466. ISBN: 978-3-540-32595-6. DOI: 10.1007/11678564_42.

2

Automated Planning of Adaptive Process Models

In this Chapter, the research questions RQ1.1 to RQ1.3 (cf. Section 1.5) are addressed. To do so, it is structured as follows. First, the automated construction of the control-flow pattern simple merge (cf. RQ1.1) will be presented in Paper 1. This paper contributes to FT1 by addressing the so far unsolved issue of constructing simple merges in an automated manner within the Automated Planning of Process Models. By aiming at constructing minimal process models (i.e., process models that do not comprise more elements than necessary), this contributes to FT2 as well, as adaptive process models comprising exclusive choices (cf. Heinrich et al., 2009; Heinrich et al., 2015b; Russell et al., 2016; van der Aalst et al., 2003) are typically not minimal. Thereafter, Papers 2 and 3 address FT2. In particular, Paper 2 strives the issue of constructing process models considering IF2 exogenous non-static context (cf. RQ1.2) and Paper 3 addresses the cooperation of IF3 different actors in an automated manner (cf. RQ1.3).

2.1 Paper 1: Automated Planning of Process Models: The Construction of Simple Merges

Status	Published	Full Citation
Accepted	06/2016	Heinrich, B., Schön, D. (2016), Automated Planning of Process Models: The Construction of Simple Merges, Proceedings of the 24th European Conference on Information Systems (ECIS 2016), June 12-15, Istanbul, Turkey.

Abstract

Business processes evolve dynamically with changing business demands. Because of these fast changes, traditional process improvement techniques have to be adapted and extended since they often require a high degree of manual work. To reduce this degree of manual work, the automated planning of process models is proposed. In this context, we present a novel approach for an automated construction of the control flow structure simple merge (XOR join). This accounts for a necessary step towards an automated planning of entire process models. Here we build upon a planning domain, which gives us a general and formal basis to apply our approach independently from a specific process modeling language. To analyze the feasibility of our method, we mathematically evaluate the approach in terms of key properties like termination and completeness. Moreover, we implement the approach in a process planning software and apply it to several real-world processes.

Keywords: Business Process Management, Process planning, Automated planning, Control Flow Structures.

Post publication changes:

- Several formatting changes for consistency reasons
- Changed citation style for consistency reasons
- The numbering of headlines, tables, figures and definitions was changed for consistency reasons
- Divide algorithm listings of getUniqueToken and addUniqueToken in Appendix
- Fixed a broken reference to a not existing “Figure 5” in Section “Operational evaluation of the results”
- Fixed a broken reference to a not existing “Section 5.2”
- Due to the availability of a printed version of Software and Systems Modeling (16.4), the year of publication of Kalenkova et al. (2017) changed
- Consistent capitalization of Section references

2.1.1 Introduction

Nowadays, as markets change, customer needs shift and new competitors evolve dynamically, companies must frequently (re)design their business processes to adapt them. At the same time, business processes span not only across departments of a single company but also across interorganizational collaborations of multiple companies, which makes process models even more complex. For instance, according to Heinrich et al. (2015b), a European bank has modeled and (re)designed over 2,000 processes in different departments and areas in a project. These process models, which are composed of actions and corresponding control flow structures, have been modeled using the ARIS toolset and documented to support upcoming improvements and adaptations of processes. To keep the process models up-to-date, frequent (re)designs due to, for instance, the aforementioned challenges of today's business world have been necessary. Moreover, the authors state that employees of the bank as well as executives of other branches such as insurance and engineering highlighted the fact that process flexibility has become more and more important within the last decade. The reasons most frequently mentioned for this increased demand for flexibility are the growing frequency and complexity of such process (re)design projects, which involve a significant degree of manual work (cf. also Hornung et al., 2007).

To ensure the required flexibility, several research fields in Business Process Management (BPM) striving to support modelers and business analysts via automatic techniques are of increasing importance. The research fields process mining as well as process model verification and validation assist the analyst in the process analysis phase (e.g., Wetzstein et al., 2007). Automated (web) service composition can be seen as part of the phases process implementation and process execution (Khan et al., 2010; Weber, 2007). In the process modeling phase, which we will focus on in this paper, the goal of the research strand automated process planning is to enable the automated construction of process models using planning algorithms (Heinrich et al., 2009; Heinrich et al., 2012; Henneberger et al., 2008; Hoffmann et al., 2012; Lautenbacher et al., 2009). Automated planning aims to increase the flexibility by definition (cf. van der Aalst, 2013) of the resulting process models and to (re)design process models - for processes that must be frequently (re)designed. The task of an automated construction of process models can be understood as a planning problem (Ghallab et al., 2004) with the objective to arrange actions and control flow structures in an appropriate order based on both, an initial state as well as a non-empty set of goal states. Here, using a nondeterministic planning domain, allowing an abstract representation of process models, independent from a specific process representation language, enables a widespread use. A fundamental challenge for the automated planning of process models is to construct control

flow structures which represent the control flow of a process (Russell et al., 2006a; van der Aalst et al., 2003). More precisely, in order to plan more complex process models, not only a sequence of actions but also control flow structures like exclusive choice, parallel split or simple merge have to be constructed in an automated manner (cf., e.g., Heinrich et al., 2009; Heinrich et al., 2015b; Hoffmann et al., 2012).

The specific research goal of this paper is the automated construction of one of the most important control flow structures, namely simple merge. The simple merge serves as a join connector for two or more paths-segments (called branch) into one single subsequent branch (cf., e.g., Russell et al., 2006a; van der Aalst et al., 2003) and thus reduces the size of process models. However, the construction of simple merges within an automated planning approach does not only focus on reducing the size of process models and thus – according to, for instance, Cardoso (2007), Mendling et al. (2010), Moreno-Montes de Oca et al. (2015), and Sánchez González et al. (2010) – its complexity. More generally, we have to be able to construct minimal process models in our context by removing redundant and duplicate path-segments "as early as possible". Further, to increase the readability and understandability of process models especially for laymen, La Rosa et al. (2011b) propose to use pattern-compounds (cf. also Gschwind et al., 2008; Mendling et al., 2007b; Mendling et al., 2010) as they represent well-formed and sound block-structured fragments of a process model. Simple merges as so-called "join connectors" (Mendling et al., 2010) therefore should only be constructed in accordance with the related "split connector" (i.e., the control flow structure exclusive choice). Following this, we aim to construct simple merges in accordance to existing exclusive choices.

The contributions of this paper are a formal definition of our planning domain and an algorithm for the automated construction of simple merges. In more detail:

❶ To follow the research field of automated process planning and thus to ensure a widespread use of our approach, we consider belief states (possibly infinite sets of world states that may exist before and after applying an action) as we address the planning of process models and thus abstract from individual process executions (cf. Ghallab et al., 2004).

❷ When constructing simple merges in an automated manner, we have to construct *minimal* process models. Thus, we address nested simple merges in order to simplify process models by removing duplicate sequences of actions in several paths and construct simple merges "as early as possible".

❸ We further focus on constructing simple merges in *complete* in terms of merging all distinct paths of process models that can be merged.

❹ We have to consider block structures (cf. Gschwind et al., 2008; La Rosa et al.,

2011a; Mendling et al., 2007a; Mendling et al., 2010) in order to increase the readability and understandability of process models by means of constructed simple merges.

In the following Section, we discuss related work regarding our contributions ❶ to ❹. Thereafter we present the formal foundation for our approach in Section 2.1.3 and the running example, we will use to illustrate our approach, in Section 2.1.4. Sections 2.1.5 and 2.1.6 elaborate the major design decisions and the proposed method to construct simple merges. In Section 2.3.5 we evaluate our approach before we conclude with a discussion, limitations of our work and an outlook to future research.

2.1.2 Related Work

Our work contributes to the research fields in BPM striving to support modelers and business analysts via automatic techniques. Especially, it focusses on the (1) automated planning of process models and is related to (2) process model complexity, (3) process modeling recommender systems and (4) process mining. Thus, we want to summarize and delimit existing research in these fields to our approach.

Ad (1): The research strand of automated planning of process models envisions the construction of process models by means of semantically annotated process elements and a semantic reasoning (Heinrich et al., 2008; Heinrich et al., 2012; Heinrich et al., 2015c; Henneberger et al., 2008; Hoffmann et al., 2012; Lautenbacher et al., 2009). Especially in Heinrich et al. (2008) and Henneberger et al. (2008) the challenges and the general planning approach is discussed. In this context, Heinrich et al. (2009) and Heinrich et al. (2015b) propose an algorithm that copes with the construction of exclusive choices based on the determination of conditions. Their approach creates conditions that enable the construction of different outgoing branches of an exclusive choice, based on co-domain of belief state variables. However, they do not cope with simple merges. In contrast, Hoffmann et al. (2012) and Hoffmann et al. (2009) discuss the need of constructing simple merges (XOR joins) within their planning. However they do not provide any kind of algorithm or implementation for this problem (we ensured this by requesting an implementation from the authors). Summing up, an approach to construct simple merges in an automated manner is not presented so far (cf. contributions ❶ to ❹).

Ad (2): Following the idea of reducing the amount of manual work through automation, several works in the field of process model complexity address the appropriate construction of control flow structures as well. Process models need to be refactored based on rules regarding the envisioned structure of process models. Therefore, control flow structures need to be transformed and constructed respectively. Vanhatalo et al.

(2008b), for example, present an approach for the automated completion of workflow graphs. Their method is based on a "well-behaved" graph with a single source (initial state) and a single sink (goal state). They aim at constructing simple merges only at the end of a process model (i.e., prior to the sink). Vanhatalo et al. (2009) and (2008) introduce the concept of refined process structure trees and a method to represent workflow graphs in such a tree-based hierarchy of sub-workflows. They aim to identify these sub-workflows but not to consolidate equal sub-workflows (cf. ❷). Polyvyanyy et al. (2011) extend this approach and present an algorithm to transform a "multi-terminal graph" (MTG), which means, a graph that has at least one source and at least one sink, to a "two-terminal graph" (TTG), which means, a graph that has exactly one source and exactly one sink. They aim to connect the existing, multiple sinks of a MTG to one common single sink of the resulting TTG, but not at consolidating equal subtrees (i.e., the representation of equal sequences of actions in different paths; cf. contributions ❷ and ❹). Munoz-Gama et al. (2014) present an approach for the decomposition of process models. Their decomposition is based on so-called "transition boundaries" or "place boundaries". That means that they identify subgraphs based on a single common action or belief state at the beginning of each subgraph. However, this is not a sufficient criterion for the construction of minimal process models and especially of nested simple merges (cf. ❷). Such nested simple merges do not necessarily require a single common *action* or *belief state* at the beginning of a subgraph. Further, none of the approaches in research field (2) copes with a nondeterministic planning domain and a state space with possibly infinite sets of world states, which is essential when addressing the automated planning of process models (cf. ❶).

Ad (3): The research strand of process modeling recommender systems focusses on issues like auto-completion of process models, finding (substructures of) process models in a repository suitable for a given problem definition or deciding where to start and stop modeling a process (cf., e.g., Fellmann et al., 2015b; Koschmider, 2007; Koschmider et al., 2011) in order to reduce the manual modeling efforts. These works aim at suggestions on correct and fitting process fragments that can be used to complete existing process models. In detail, during the construction of process models, recommender systems propose fitting process fragments (saved in a process model repository) based on (semantic) similarity measures of the fragments and the given problem definition, represented by, for instance, incomplete constructed process models at hand. This promising work, however, does not aim on constructing simple merges in an automated manner (cf. especially ❶ to ❸).

Ad (4): Besides these approaches, process mining aims at the partially automated reconstruction and redesign of process models based on event logs. Process mining allows discovering, checking and enhancing process models including workflow patterns

(cf., e.g., Gaaloul et al., 2005a) by means of event logs (cf., e.g., Accorsi et al., 2012; van der Aalst et al., 2012). As (van der Aalst et al., 2012) explicate, process mining should support basic control flow structures. The authors stated that existing algorithms like the alpha algorithm (cf., e.g., Gaaloul et al., 2005a; Gaaloul et al., 2005b; van der Aalst et al., 2004) are able to construct simple merges. However, these algorithms follow a local perspective by examining pairwise relations between two actions. Thus, they do not aim to construct simple merges *in complete* (cf. ❸) and may not provide minimal process models (cf. ❷). Further, to the best of our knowledge, we found no approach to construct simple merges in an automated manner that considers block structures (cf. ❹) for increasing the readability of the constructed process models. Moreover, current conversion algorithms (cf., e.g., Kalenkova et al., 2017), that translate Petri Nets into Process Models (here: BPMN), do not deal with the completeness (cf. ❸) of the constructed simple merges. Moreover, as process mining aims to *re*construct *as-is* process models based on event logs, it does not cope with the ex-ante construction of *to-be* process models as it is addressed in this paper. Further, the field of process mining usually does not cope with a nondeterministic planning domain and a state space with possibly infinite sets of world states. So, the approaches for the construction of workflow patterns (as stated in e.g., Gaaloul et al., 2005b; van der Aalst et al., 2010), used in process mining, do not aim to address contribution ❶. To sum up, to the best of our knowledge, there exists no approach that addresses all contributions ❶ to ❹.

2.1.3 Fundamentals

As stated above, the construction of simple merges is a nondeterministic planning problem with belief states because we abstract from an individual process execution (Ghallab et al., 2004). Using a nondeterministic planning domain which is independent from a particular process representation language enables a widespread use and guarantees compatibility with many existing approaches in the literature (e.g., Bertoli et al., 2001; Bertoli et al., 2006). A nondeterministic planning domain consists of a nondeterministic belief state-transition system which is defined in terms of its belief states, its actions, and of a transition function that describes how (the application of) an action leads from one belief state to possibly many belief states (acc. Bertoli et al., 2006; Ghallab et al., 2004). More formally, a belief state-transition system and (non-)determinism in state space are defined as follows:

Definition 2.1.1. (*Nondeterministic state-transition system*). A nondeterministic belief state-transition system is a tuple $\Sigma = (BS, A, R)$, where

- BS is a finite set of belief states. A belief state $bs \in BS$ contains a set BST of belief state

tuples. A belief state tuple p is a tuple of a belief state variable $v(p)$ and a subset $r(p)$ of its predefined domain $dom(p)$, which we will write as $p{:=}(v(p),r(p))$.

- A is a finite set of actions. Each action $a{\in}A$ is a triple consisting of the action name and two sets, which we will write as $a{:=}(name(a), precond(a), effects(a))$. The set $precond(a){\subseteq}BST$ are the preconditions of a and the set $effects(a){\subseteq}BST$ are the effects of a.

- And $R{:}\ BS{\times}A \rightarrow 2^{BS}$ is the transition function. The transition function associates to each belief state $bs{\in}BS$ and to each action $a{\in}A$ the set $R(bs, a){\subseteq}\ BS$ of next belief states.

According to this definition it is possible to represent possibly infinite sets of world states quite easily. Furthermore, it is a rather intuitive way – from a process modeling perspective – to represent certain preconditions and effects of actions.

Definition 2.1.2. ((*Non-*)*determinism in state space*). An action a is *applicable* in a belief state bs bs iff $|R(bs, a)| > 0$; it is *deterministic* (*nondeterministic*) in bs bs iff $|R(bs, a)| = 1$ ($|R(bs, a)| > 1$). If a is applicable in bs, then $R(bs, a)$ is the set of belief states that can be reached from bs by performing a.

Based on both Definitions 2.1.1 and 2.1.2, a planning graph can be generated by means of different existing algorithms that progress from an initial belief state to goal belief states (see, e.g., Bertoli et al., 2001; Bertoli et al., 2010; Heinrich et al., 2009; Heinrich et al., 2012). In this paper, we primarily are extending these works by means of an approach to construct simple merges in an automated manner. With that said, we define our planning graph as follows:

Definition 2.1.3. (*planning graph*). A planning graph is an acyclic, bipartite, directed graph $G = (N, E)$, with the set of nodes N and the set of edges E. Henceforth, the set of nodes N consists of two partitions: First, the set of flow nodes $Part_F$ (set F of flow nodes) which further contains two partitions, the set of action nodes $Part_A \subseteq Part_F$ (set A of actions) and the set of exclusive choice nodes $Part_{EC} \subseteq Part_F$ (set EC of exclusive choices), and second the set of belief state nodes $Part_{BS}$(set BS of belief states). Each node $bs \in Part_{BS}$ is representing one distinct belief state in the planning graph. Each node $a{\in}Part_A$ is representing an action in the planning graph. Each node $ec{\in}Part_{\mathrm{EC}}$ is representing an exclusive choice node in the planning graph. The planning graph starts with one initial belief state and ends with one to probably many goal belief states (with $Init{\in}BS$ and $Goal_{\mathrm{j}}{\in}BS$).

When focusing on automated process planning, identical or very similar actions of a planning graph – as specified in Definition 2.1.3 – can be identified by using semantic

concepts and automated reasoning (cf. e.g., Step 1, Heinrich et al., 2015b, p. 3). Such actions can then be identically (syntactically) labelled in the planning graph, which we will use in the following. However, our approach is not limited to the strand of automated process planning. In fact, for instance, within the research field of process mining, several works exist (see, e.g., Kindler et al., 2006; van der Aalst et al., 2010; Verbeek et al., 2007) that use or construct graphs similar to planning graphs. Moreover, we envision to apply our approach to manually constructed process models by transferring them to the notions of a planning graph.

Given Definition 2.1.3, a planning graph consists of one to many paths. Here, a path is defined as follows:

Definition 2.1.4. (*path*). A path is sequence of nodes $n_{\mathrm{p}} \in N$ starting with the initial belief state and ending in exactly one goal belief state.

Further, we define a branch as a subset of nodes regarding a particular path, starting with an *exclusive choice* node $ec \in Part_{\mathrm{EC}}$:

Definition 2.1.5. (*branch*). A branch is a sequence of nodes $n_{\mathrm{b}} \in N$ starting right after an exclusive choice node $ec \in Part_{\mathrm{EC}}$ and ending in exactly one goal belief state. A branch therefore is a subset of nodes of a specific path.

2.1.4 Running Example

We will use an excerpt of a real-world process taken from the order management of a European bank to illustrate our approach in the following. This excerpt of an order execution process model is represented by the planning graph (such a graph can be constructed with one of the approaches proposed by Bertoli et al., 2006; Heinrich et al., 2009; Heinrich et al., 2015b) shown in Figure 2.1. As illustrated, the order data must be entered and determined in a first step. After that, depending on the type of the security (conditions), a check must be stated and the order amount must be entered or calculated (in case of a stock order). Then, the plausibility of the order has to be proven and the order will get executed before the order is processed internally or externally and has to be assigned to a portfolio and documented. Finally, the order gets routed (note: the uppercase letters A-G are used to refer to the according XORs hereafter).

2.1.5 Design Process

In this Section we will outline the major design decisions to address the contributions ❶- ❹. These major design decisions are as follows:

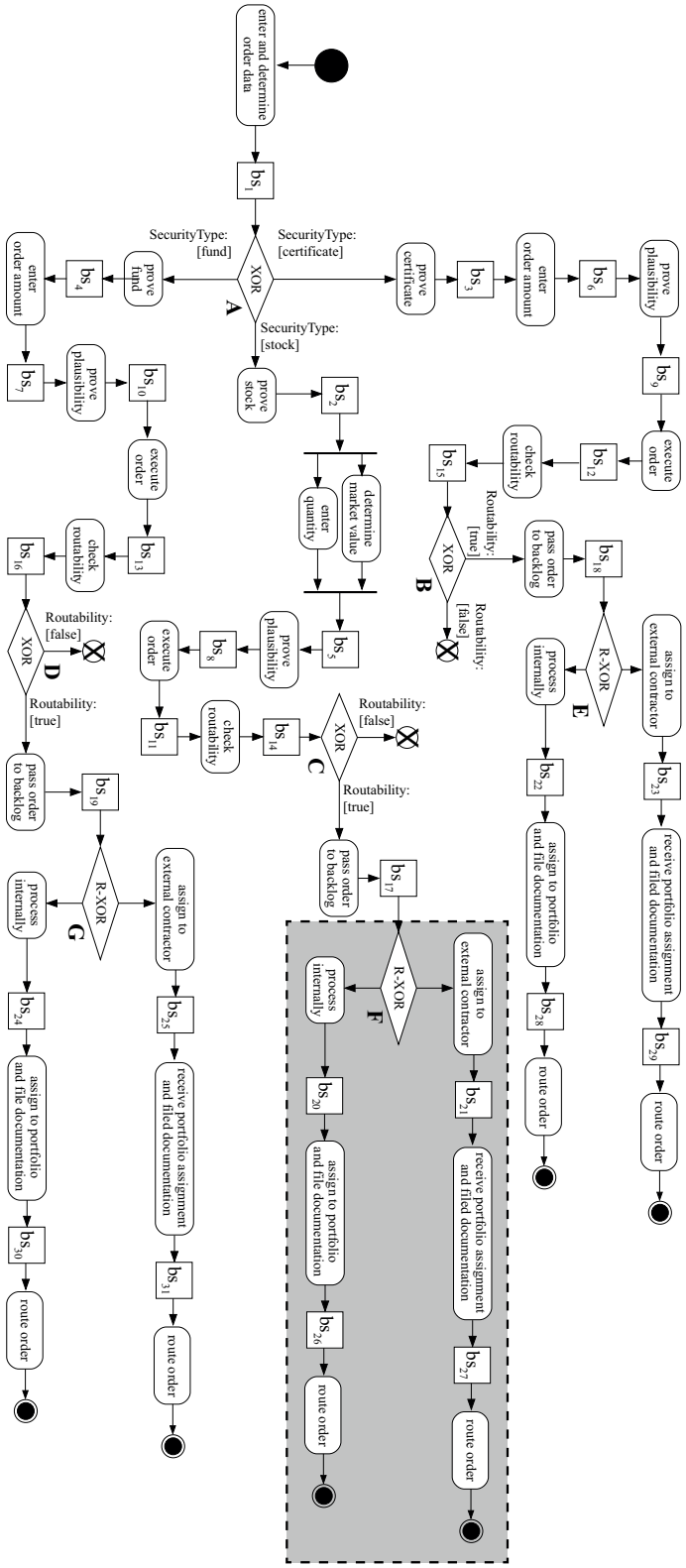

Figure 2.1: Planning Graph of our Running Example

1. *Traverse backwards:* To construct simple merges, we traverse the planning graph backwards, starting from the goal belief states. As the planning graph ends with one to many goal belief states, it is necessary to start the traversal with these goal belief states to identify all potentially mergeable paths and thus to cope with ❷ and ❸. The reason for traversing the planning graph backwards is that we are able to identify mergeable paths directly and do not need to traverse them completely from the initial state (like if we would use a forward traversal). Precisely, traversing backwards is effective because paths that can be combined by a simple merge need to *end* in both equal actions and equal control flow structures.

2. *Mark flow nodes with tokens:* To identify mergeable paths, we compare the flow nodes (partition $Part_{\mathrm{F}}$) of the paths in a breadth-first manner, i.e., all flow nodes preceding the goal belief states are compared in the first iteration, all flow nodes preceding these in the second and so on. This approach, combined with 1), allows us to identify all mergeable paths directly (cf. ❷ and ❸), as they would be equal from a specific action in the path until the goal belief state. We use tokens to mark sets of equal flow nodes being part of different paths. For instance, given that the last flow node before the goal belief state is equal in three paths, we mark these flow nodes with the same token, otherwise with different ones. Continuing traversing backwards and annotating actions with tokens breadth-first allows us to recognize potentially mergeable paths even if they differ in subsequent iterations.

3. *Construct simple merges:* To assure that paths are merged only if they were split by an exclusive choice previously (cf. ❹), we construct simple merges when we reach an exclusive choice in the backward traversal. Here, the annotated tokens are used to identify mergeable paths.

Addressing design decision 2), we have to handle the need of identifying equal sequences of flow nodes within different paths. Thus, we define so-called *equality groups*:

Definition 2.1.6. (*equality group*). An *equality group* $eg_t = (n_1, \ldots, n_m)$ is a tuple of flow nodes with the following properties:

- $n_i \, \epsilon$ Part$_F$ for all $i \, \epsilon \, \{1, \ldots, m\}$
- n_{i+1} succeeds n_i for all $i \, \epsilon \, \{1, \ldots, m-1\}$
- A token T_t denotes a flow node as member of the equality group eg_t

We denote the set of all equality groups by EG .

2.1.6 Method to Construct Simple Merges

In this Section, we will elaborate and employ the above mentioned major design decisions to design a method that constructs simple merges and addresses the contributions ❶ to ❹.

2.1.6.1 Step 1: Merging one single Exclusive Choice

In the first step, we address to merge - as early as possible - two or more outgoing branches regarding a single exclusive choice. In this simple case, we conduct the following sub steps:

1. Check all paths, starting from their corresponding goal belief states. Mark equal flow nodes with the same *equality token*, starting with the last flow node before each goal belief state.

2. Continue traversing backwards, comparing the subsequent flow nodes of all paths. If the flow nodes of each considered equality group are still equal, simply add the same token to the flow nodes of these paths. If the flow nodes are different, add new tokens for each equality group of flow nodes. Consider that flow nodes could only be marked with the same additional token, if they are member of the same equality group so far, which means if their preceding (in the backward traversal) flow nodes are the same. We additionally add the tokens of previously marked flow nodes to each flow node for performance reasons with regard to later phases of our approach.

3. If an exclusive choice is reached while traversing the branches backwards, we need to check the first flow node of all outgoing branches. Those branches that are marked with the same *equality token* are mergeable as they contain an equal sequence of flow nodes. Thus, they have to be merged with a *simple merge* just before the beginning of this equal sequence, which we will recognize as all contained actions are marked only with equal tokens. Further, the belief states of the merged branches have be joined (regarding our running example, the set union of $SecurityType : fund$, $SecurityType : certificate$ and $SecurityType : stock$) to get the accurate belief state regarding the single subsequent path after the simple merge. If the branches are not marked with equal tokens they will be merged only before the goal belief state. The reached exclusive choice itself is marked with a new token and all common tokens of the merged branches, too.

4. Note: If branches of an exclusive choice have different lengths, all branches, except the longest, stop the backward traversal at the exclusive choice. When the traver-

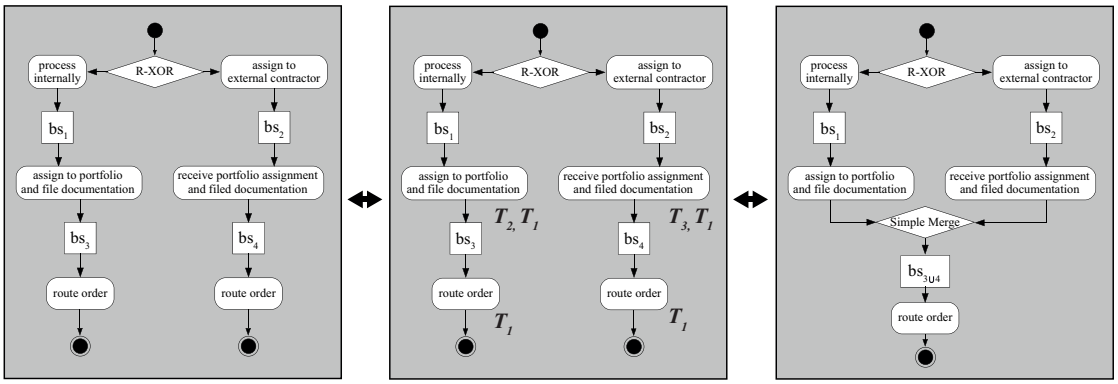

Figure 2.2: Constructing a Simple Merge regarding the Running Example

sal of the longest branch reaches the exclusive choice, all outgoing branches are compared according to the upper description.

5. Finally, we have to continue traversing backwards and marking all flow nodes with tokens. Further, the approach finally reaches the initial state and terminates.

To demonstrate this first step, we use the introduced order execution process model. For illustration purpose, we focus on the part of the process, which is framed in Figure 2.1. According to the example, both paths are marked with the token T_1, as both paths end with the equal action "route order" (sub step (1)). Further, the actions "assign to portfolio and file documentation" and "receive portfolio assignment and filed documentation" differ and hence are additionally marked with different tokens (T_2 resp. T_3) according to sub step (2). Then, reaching the exclusive choice, the outgoing branches are identified as mergeable, as both contain the token T_1. Thus, a simple merge is inserted before the action "route order", as this is the first flow node marked only with equal tokens in both branches (see rightmost graph in Figure 2.2). Further, all subsequent belief states (i.e., bs_3 and bs_4) have be joined.

2.1.6.2 Step 2: Merging multiple nested Exclusive Choices

When process models become more complex, exclusive choices could occur in branches of other exclusive choices etc. Thus, our approach must be able to cope with *nested* exclusive choices (e.g. exclusive choices E, F and G are nested in exclusive choice A; cf. Figure 2.1). To address this challenging problem, we need to consider the main components of an exclusive choice, the outgoing branches and their conditions, and define a *choice construct* as follows:

Definition 2.1.7. (*choice construct*). The choice construct C of an exclusive choice is

defined as a set of $c(c \in C)$ where c is defined for each outgoing branch of this exclusive choice. Moreover, c is defined as the tuple $c := (Condition, Tokens)$ consisting of the condition of the corresponding branch (or null if the branch has no conditions) and the tokens of the first action of the branch succeeding the exclusive choice.

Based on this definition we are able to cope with *multiple nested* exclusive choices. To enable the merging of several paths containing exclusive choices with equal choice constructs – and therefore with equal outgoing branches and conditions – we need to *extend* the above sub step (3) as follows:

(3′) As defined above in sub step (3), when reaching an exclusive choice, merge the outgoing branches. Additionally, mark the exclusive choice with its choice construct. Compare the exclusive choice with every other exclusive choice based on their choice constructs similar to comparing actions and mark them with the same token, when two or more choice constructs are equal.

The general specification in the first step above in combination with the straightforward extensions of Definition 2.1.7 and sub step (3′) allows us to cope with the problem of merging multiple *nested* exclusive choices in outgoing branches now. For further comparison the tokens of the outgoing branches of a nested exclusive choice are not required anymore and thus, when marking subsequent flow nodes, only the token of the exclusive choice is needed.

To illustrate these extensions, we consider the whole planning graph of our running example as seen in Figure 2.1, containing multiple exclusive choices (denoted by capital letters). Initially, the procedure is equal to the excerpt in Figure 2.2 until the exclusive choices B, C and D are reached. When reaching each of the exclusive choices E, F and G, both outgoing branches are marked with the equality tokens T_1 and thus are mergeable before the action "route order". Further, exclusive choice F is marked with its choice construct, precisely

$$(null, (T2, T1)), (null, (T3, T1))$$

(cf. sub step (3′)). As exclusive choices E and G contain the same outgoing branches as F, they are marked with the same choice construct. According to sub step (3′), all of those three exclusive choices get marked with T_4, as their choice constructs are equal.

When continuing the backward traversal, the exclusive choices B, C and D are reached and compared. They have to be marked with

$$\{((Routability, false), null), ((Routability, true), (T4, T1))\}$$

and their outgoing branches can be merged. As there is only one outgoing branch that

leads to the goal[1], no further merge is needed. Thereafter (cf. sub step (4)), the outgoing branches of exclusive choice A are checked. As they all contain the equal token T_5 (cf. the equal exclusive choices B, C and D) they will be merged before the action "prove plausibility" resulting in the graph shown in Figure 2.4.

2.1.6.3 Step 3: Merging Exclusive Choices within Parallelization Compounds

Regarding Mendling et al. (2010), process models should be "as structured as possible" (i.e., "every split connector matches a respective join connector of the same type"), which is related to contribution ❹. So far, the proposed approach is not able to provide this in case an exclusive choice is created within a parallelization compound (i.e., path segments surrounded by a parallel split node and a synchronization node). To solve this issue, the Definitions 2.1.3 to 2.1.5 must be extended to allow the representation of planning graphs containing parallelization compounds:

Definition 2.1.8. (*planning graph*). A planning graph is an acyclic, bipartite, directed graph $G=(N, E)$, with the set of nodes N and the set of edges E. Henceforth, the set of nodes N consists of two partitions: First, the set of flow nodes $Part_{\mathrm{F}}$ (set F of flow nodes) which further contains four partitions, the set of action nodes $Part_{\mathrm{A}} \subseteq Part_{\mathrm{F}}$ (set A of actions), the set of exclusive choice nodes $Part_{\mathrm{EC}} \subseteq Part_{\mathrm{F}}$ (set EC of exclusive choices), the set of parallel split nodes $Part_{\mathrm{PS}} \subseteq Part_{\mathrm{F}}$ (set PS of parallel splits) and the set of synchronization nodes $Part_{\mathrm{S}} \subseteq Part_{\mathrm{F}}$ (set S of synchronizations), and second the set of belief state nodes $Part_{\mathrm{BS}}$ (set BS of belief states). Each node $bs \in Part_{\mathrm{BS}}$ is representing one distinct belief state in the planning graph. Each node $a \in Part_{\mathrm{A}}$ is representing an action in the planning graph, each node $ec \in Part_{\mathrm{EC}}$ is representing an exclusive choice node in the planning graph, each node $p \in Part_{\mathrm{PS}}$ is representing a parallel split node in the planning graph and each node $s \in Part_{\mathrm{S}}$ is representing a synchronization node in the planning graph. The planning graph starts with one initial belief state and ends with one to probably many goal belief states (with $Init \in BS$ and $Goal_{\mathrm{j}} \in BS$).

Definition 2.1.9. (*path*). A path is sequence of nodes $n_{\mathrm{p}} \in N$ either 1) starting with the initial belief state and ending in exactly one goal belief state or 2) starting with a parallel split node $p \in Part_{\mathrm{PS}}$ and ending in the corresponding synchronization node $s \in Part_{\mathrm{S}}$.

Definition 2.1.10. (*branch*). A branch is a sequence of nodes $n_{\mathrm{b}} \in N$ starting right after an exclusive choice node $ec \in Part_{\mathrm{EC}}$ and ending in exactly one goal belief state or a synchronization node $s \in Part_{\mathrm{S}}$. A branch therefore is a subset of nodes of a specific path (see Definition 2.1.9).

[1] The other outgoing branch leads to a so-called flow final node, which terminates the process and is not needed to be merged.

Now, we are able to represent process models containing parallelization compounds by means of our planning domain. The upper part of Figure 2.3 shows an extension of our running example illustrating this. To enable the automated construction of simple merges within parallelization compounds we carefully extend the previously presented sub steps (1), (3) resp. (3′) and (4) as follows:

(3″) When reaching a synchronization node, invoke the overall method (i.e., sub steps (1_{para}), (2), (3), (3′), (4_{para})) for all paths between the parallel split node and the synchronization node.

(1_{para}) Check all paths, starting from the final synchronization node. Mark equal flow nodes with the same *equality token*, starting with the last flow node before the synchronization node.

(4_{para}) Continue to traverse backwards and mark all flow nodes with tokens. Further, when finally reaching the initial parallel split node, return to the enclosing iteration.

We follow the above presented idea, used for merging nested exclusive choices, to allow merging branches that contain *nested* parallelization compounds and define a so-called *parallel construct* based on the contained actions:

Definition 2.1.11. (*parallel construct*). A parallel construct P of a parallel split is defined as a set of T_t ($Tt \in P$) where T_t is denoted for each outgoing branch of this parallel split. Moreover, T_t is defined as the token of the first flow node of the branch succeeding the parallel split.

Further, when merging a nested parallelization compound, we compare it with every other parallelization compound in the currently analyzed equality group based on their parallel constructs (cf. sub step (2)) and mark them with the same token, when two or more parallel constructs are equal.

Regarding our running example (Figure 2.3), we traverse the parallelization compound backwards, starting with the synchronization node as stated in (1_{para}). Thereupon, "enter quantity" could be identified as equal in the two branches of the exclusive choice in sub step (2). When continuing the traversal and finally reaching the exclusive choice, these two branches will be merged as seen in the lower area of Figure 3 regarding (3). Regarding the previously defined sub steps (1) to (4), the parallel split will be marked with the parallel construct T_2, T_4 when finishing the traversal of the parallelization compound (cf. (4_{para})), as "determine real-time market value" gets marked with T_2 and "determine budget" gets marked with T_4. To sum up, we are now able to create minimal and block-structured (cf., e.g., La Rosa et al., 2011c; Mendling et al., 2010) process models by constructing simple merges in complete.

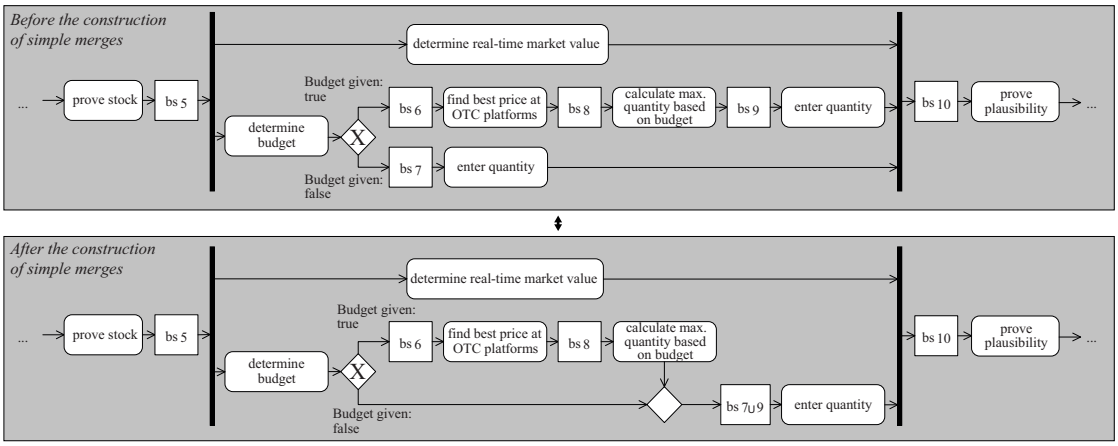

Figure 2.3: Unmerged and Merged Exclusive Choice within a Parallelization Compound

2.1.7 Evaluation

In this Section, we sketch the formal evaluation of our approach by mathematically proving that it provides minimal process models (i.e., simple merges are constructed "as early as possible"; cf. ❷), constructs simple merges if possible (i.e., completeness; cf. ❸) and terminates. We further briefly sketch its computational complexity and evaluate the results of our approach with respect to the construction of block structures (cf. ❹). Afterwards we evaluate the feasibility of the approach by means of a prototypical implementation and applying it to several real-world processes.

2.1.7.1 Formal Evaluation of the Approach

For the formal evaluation of our approach, we need to ensure that it terminates, identifies all mergeable paths (completeness), does not construct incorrect simple merges (correctness) and constructs simple merges as early as possible (i.e., is minimal). It is proven, that the approach meets all these criteria and its computational complexity is $O(n^5)$ in the number of goal belief states or sequential flow nodes of the longest path of the planning graph. This means, the algorithm is computationally efficient (cf. Arora et al., 2009; Cobham, 1965). It is further proven that for each exclusive choice, exactly one simple merge is created (cf. ❹) so that the results of the approach are syntactically correct and fulfill the criteria of soundness and s-coverability. For the proof sketches see Appendices 7.1.2 and 7.1.3.

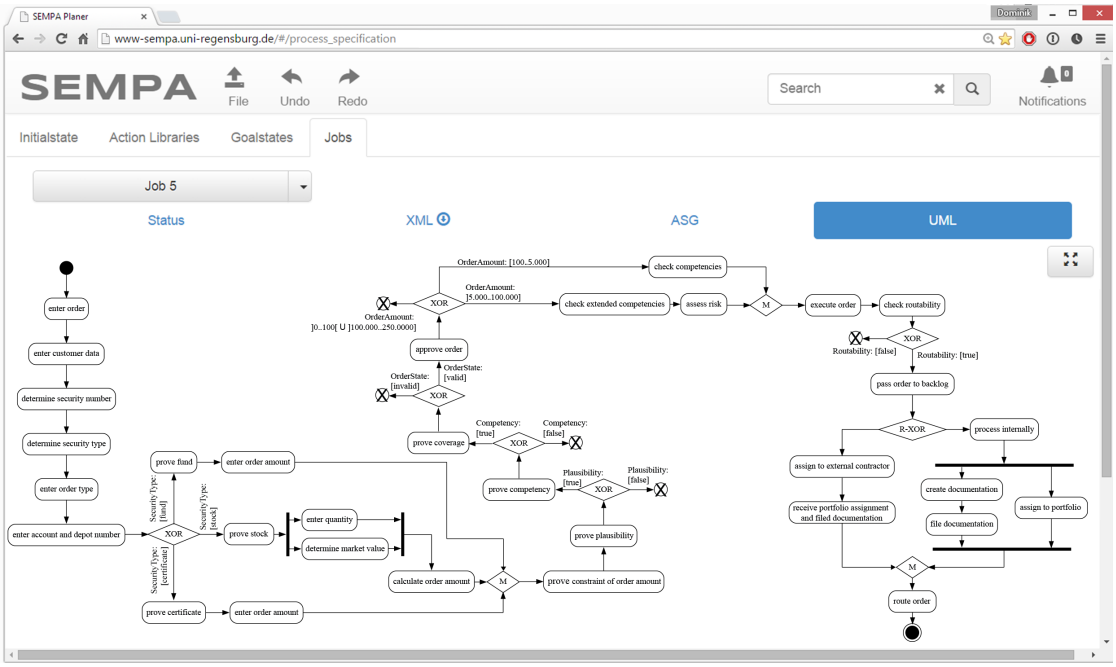

Figure 2.4: Screenshot of the Constructed Process Model by means of our Prototype

2.1.7.2 Operational Evaluation of the Results

To assess the feasibility and applicability of our approach, the following questions are evaluated:

(E1) Can our approach be instantiated in terms of a prototypical software implementation?

(E2) Can it be applied in a practical setting and what is the output resulting from its application?

We integrated our approach in a web-based process planning tool (cf. E1; a demo version could be accessed at `http://www-sempa.uni-regensburg.de/`). The web application guides the user while populating the set of actions by providing descriptions of preconditions and effects and specifying the initial state and goal states. To test the implementation, persons other than the programmers analyzed the source code and several extreme value tests and unit tests have been performed. The implementation did not show any errors at the end of the test phase.

By means of the software implementation, we applied our approach to several real-world processes of a European bank, European insurance companies and an educational institution (cf. E2). Our running example is a part of one of these processes and ad-

dresses the order management of the bank (cf. Figures 2.1 and 2.4). To be able to apply our algorithm, the actions of former process models, used in the area of security order management had to be extracted and afterwards imported in our planning tool. About 200 actions including their preconditions and effects could be imported via an XML interface of ARIS, the bank's process modeling tool. We then reviewed them to validate, for instance, that their preconditions and effects were accurate noted. Finally, we specified the initial state and the intended goal states with the help of the employees of the bank's responsible department and planned the feasible process models using our planning tool. Hereby, two simple merges are constructed. The simple merge node that is firstly constructed merges the branches after "receive portfolio assignment and filed documentation" and "assign to portfolio and file documentation". Its construction is straightforward as no nested exclusive choices need to be created. In the next step, the outgoing branches of the exclusive choices B, C and D are analyzed. As here only one branch leads to the goal and the other branch leads to a flow final node, no further simple merge needs to be created. As the previously merged exclusive choice occurs in this branch again, it is needed to consider the extensions for merging nested exclusive choices (cf. Section 2.1.6.2). In the last step, a simple merge node before "prove plausibility" merges the three different branches depending on the security type. Here, the extensions for nested exclusive choices (cf. Section 2.1.6.2) have to be considered again. In conclusion, only one action ("enter order amount") appears twice in the process model due to respecting pattern compounds (cf. contribution ❹).

As stated above, we applied the approach to further processes in different contexts of different firms. Table 2.1 shows the results of these applications (executed on an Intel Core i7-2600 3.40 GHz, Windows 7 64 Bit, Kernel Version 7601.22616, Java 8). The eleven analyzed processes of different application contexts include up to 278 actions and 189 states in the initial graph and are therefore of a small to a large size. The largest process model No. 10, for instance, contains actions conducted by several departments of the European bank and external service providers. Other processes are run by an insurance company, a mechanical engineering company (the context "Project management") and a university (the context "Human resources"). All process models include simple merges and for many process models nested control flow structures were created that merge a significant number of actions and states. This illustrates that nested simple merges are frequently used and relevant control flow structures. In all situations, our approach was fully applicable and generated correct and complete solutions. Thus the practical applicability of the approach is supported. The run time to construct simple merges varies from 0.001 up to 5.405 sec. and is thus very small.

Regarding economic aspects, Krause et al. (2013) present a quantitative evaluation, analyzing 18 process modeling projects from a financial service provider. Here, auto-

Process number	Context	Num. of actions / states in the initial search graph	Num. of constructed simple merges	Num. of actions included by the simple merge	Num. of actions and control flow structures merged by the simple merge	Num. of merged paths	Run time
1	Project Mgmt.	17/15	1	4	1/0	3	0.001
2	Project Mgmt.	25/18	1	1	7/2	2	0.001
3	Project Mgmt.	26/22	3	8	8/1	4	0.002
4	Project Mgmt.	38/25	2	6	36/3	4	0.001
5	Insurance Mgmt.	43/38	6	29	19/7	27	0.016
6	Insurance Mgmt.	54/44	8	33	18/8	3	0.006
7	Loan Mgmt.	40/31	3	25	12/3	3	0.003
8	Loan Mgmt.	57/43	4	14	11/4	20	0.013
9	Loan Mgmt.	122/69	3	54	4/0	0	0.024
10	Private Banking	278/189	25	82	69/16	6772	5.405
11	Human Res.	83/75	10	76	3/0	4	0.016

Table 2.1: Application of our Approach in further Real-use Situations

mated planning generates higher initial setup costs than manual process modeling, especially for analyzing and annotating actions. In contrast, the ongoing modeling costs are (as expected) much lower because of the support by the automated planning approach. Moreover, Krause et al. (2013) show that the use of automated planning of process models increases the contribution margin by about 20% which should cover its necessary higher initial investments. Automated planning should likely be even more valuable over a long term as Krause et al. (2013) considered only a short period of time. Alongside with this findings, applying the presented approach allows reducing the manual efforts when constructing process models and thus reduces the variable costs within the planning process. Hence it is supported that our approach is even more valuable for process models that need to be redesigned frequently as the initial costs can be amortized by savings of parts of the variable costs that occur with each redesign (cf. also Heinrich et al., 2015b). Furthermore, the complexity of the process model could be decreased by reducing the amount of nodes within the process model, which then may lead to less errors during the execution of the process (cf. La Rosa et al., 2011c; Laue et al., 2010; van der Aalst et al., 2008).

2.1.8 Discussion and Conclusion

We propose a novel approach to construct the control flow structure simple merge in an automated manner and thus contribute to the research strand of automated process planning. Further, this work aligns to several research fields in BPM striving to support modelers and business analysts via automatic techniques. To abstract from individual process executions and to ensure a widespread use of our approach, we consider belief states (cf. ❶). We construct minimal (cf. ❷) and complete (cf. ❸) process models. Within this paper we additionally considered the construction of pattern compounds as proposed by e.g. La Rosa et al. (2011c) in order to increase readability and understandability of the process models (cf. ❹). Our approach and the planning domain are formally noted and can therefore be well-defined and evaluated by means of mathematical proofs. This guarantees that key properties and envisioned contributions are met. Further, we discussed its applicability, feasibility and the results from the practical application by applying our approach (implemented in a process planning tool) to several real-world processes. In this context, we have tested that the construction of simple merges contributes to the automated planning of entire process models and thereby to reduce the amount of manual efforts within process (re-)design.

However, our research has some limitations that need to be addressed in the future. Constructing block structures (cf. ❹) may imply redundancies in the resulting process model that will not be merged in some cases (cf. action “enter order amount” in our

running example). In some cases this might not be favorable, for instance, if the resulting process models are used only by domain experts in process modeling. In future research, this issue has to be addressed. To enable this, choice constructs of nested exclusive choices have to be compared in depth instead of only using their token to identify mergeability.

To complete the automated planning of entire process models, further (advanced) control flow structures should be constructed. Moreover, the work of Krause et al. (2013) should be followed-up to ensure a valuable usage of automated process planning in future real-world cases. For instance, it should be evaluated whether automated process planning makes it easier for laymen to construct correct and feasible process models. Further, it should be evaluated how the presented approach could be applied in other research strands such as process mining. As process mining approaches usually employ event logs to derive process descriptions (e.g., process graphs), we expect our approach to be beneficial especially regarding contributions ❷ and ❸ in this research strand, too. However, this idea needs to be evaluated in future research.

Further, it should be evaluated how process models modeled in a manual manner could be enriched to enable transferring them to planning graphs. Additionally, combined with assessing the semantic similarity of actions, our approach seems promising for supporting modelers as planning approaches (cf., e.g., Bertoli et al., 2006; Heinrich et al., 2008; Heinrich et al., 2012) are already based on semantic annotations. For assessing the similarity of actions and subgraphs, works in the research fields of process management (cf., e.g., Ehrig et al., 2007; Minor et al., 2007; Montani et al., 2015) and web service composition can be applied as especially the latter ones use input and output parameters as we do (cf., e.g., Dong et al., 2004). Such works should be useable in the context of automated planning of process models as well. However, actions that are similar but not equal may not be merged in an automated manner but it might be possible to suggest modelers which actions may be merged after a modification. Such an extension of our approach provides a basis for promising advancements in the future.

2.1.9 Acknowledgements

The research was funded by the Austrian Science Fund (FWF): P 23546-G11.

2.1.10 References

Accorsi, R., Ullrich, M., and van der Aalst, W. M. P. (2012). "Aktuelles Schlagwort: Process Mining." In: *Informatik Spektrum* 35.5, pp. 354–359.

Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bertoli, P., Pistore, M., and Traverso, P. (2010). "Automated composition of Web services via planning in asynchronous domains." In: *Artificial Intelligence* 174.3-4, pp. 316–361. ISSN: 0004-3702. DOI: 10.1016/j.artint.2009.12.002.

Cardoso, J. (2007). "Complexity analysis of BPEL Web processes." In: *Software Process: Improvement and Practice* 12.1, pp. 35–49. ISSN: 10774866. DOI: 10.1002/spip.302.

Cobham, A. (1965). "The intrinsic computational difficulty of functions." In: *Logic, Methodology, and Philosophy of Science II*.

Dong, X., Halevy, A., Madhavan, J., Nemes, E., and Zhang, J. (2004). "Similarity search for web services." In: *Proceedings of the Thirtieth International Conference on Very Large Data Bases*. Vol. 30. St. Louis, MO: Morgan Kaufmann, pp. 372–383. ISBN: 0-12-088469-0.

Ehrig, M., Koschmider, A., and Oberweis, A. (Feb. 2007). "Measuring Similarity between Semantic Business Process Models." In: *Proceedings of the 4th Asia-Pacific Conference on Conceptual Modelling (APCCM 2007)*. Ed. by J. F. Roddick and A. Hinze. Vol. 67. Ballarat, Victoria, Australia, pp. 71–80.

Fellmann, M., Zarvic, N., Metzger, D., and Koschmider, A. (2015b). "Requirements Catalog for Business Process Modeling Recommender Systems." In: *Wirtschaftsinformatik Proceedings 2015*. Ed. by O. Thomas and F. Teuteberg, pp. 393–407.

Gaaloul, W., Alaoui, S., Baïna, K., and Godart, C. (2005a). "Mining Workflow Patterns through Event-data Analysis." In: *Proceedings of the The 2005 Symposium on Applications and the Internet Workshops (SAINT-W'05)*, pp. 226–229.

Gaaloul, W., Baïna, K., and Godart, C. (2005b). "Towards Mining Structural Workflow Patterns." In: *Database and Expert Systems Applications*. Ed. by K. Andersen, J. Debenham, and R. Wagner. Vol. 3588. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 24–33. ISBN: 978-3-540-28566-3. DOI: 10.1007/11546924_3. URL: http://dx.doi.org/10.1007/11546924_3.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Gschwind, T., Koehler, J., and Wong, J. (2008). "Applying Patterns during Business Process Modeling: Business Process Management." In: *Business Process Management*. Ed.

by M. Dumas, M. Reichert, and M.-C. Shan. Vol. 5240. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 4–19. ISBN: 978-3-540-85757-0. DOI: 10.1007/978-3-540-85758-7_4.

Heinrich, B., Bewernik, M.-A., Henneberger, M., Krammer, A., and Lautenbacher, F. (2008). "SEMPA - A Semantic Business Process Management Approach for the Planning of Process Models." In: *Business & Information Systems Engineering (formerly Wirtschaftsinformatik)* 50.6, 445–460 (in German). URL: http://epub.uni-regensburg.de/23173/.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems (ICIS)*. Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Heinrich, B. and Schön, D. (2015c). "Automated Planning of context-aware Process Models." In: *Proceedings of the 23rd European Conference on Information Systems (ECIS)*. Ed. by J. Becker, J. vom Brocke, and M. de Marco. Münster, Germany, Paper 75.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik (MKWI)*. Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hoffmann, J., Weber, I., and Kraft, F. M. (2009). "Planning@ sap: An application in business process management." In: *Proceedings of the 2nd International Scheduling and Planning Applications woRKshop (SPARK'09)*.

Hoffmann, J., Weber, I., and Kraft, F. M. (July 2012). "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management." In: *Journal of Artificial Intelligence Research* 44.1, pp. 587–632. DOI: 10.1613/jair.3636.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference*

on Advanced Information Systems Engineering (*CAiSE*). URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

Kalenkova, A. A., van der Aalst, W. M. P., Lomazova, I. A., and Rubin, V. A. (Oct. 2017). "Process Mining Using BPMN: Relating Event Logs and Process Models // Process mining using BPMN: Relating event logs and process models." In: *Software and Systems Modeling* 16.4, pp. 1019–1048. DOI: 10.1007/s10270-015-0502-0.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security* (*IJCSIS*) 7.2, pp. 147–152.

Kindler, E., Rubin, V., and Schäfer, W. (2006). "Process Mining and Petri Net Synthesis." In: *Business Process Management Workshops*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Eder, and S. Dustdar. Vol. 4103. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 105–116. ISBN: 978-3-540-38444-1. DOI: 10.1007/11837862_12.

Koschmider, A. (2007). *Ähnlichkeitsbasierte Modellierungsunterstützung für Geschäftsprozesse*. Karlsruhe: Univ.-Verl. Karlsruhe. ISBN: 978-3-86644-188-0.

Koschmider, A., Hornung, T., and Oberweis, A. (2011). "Recommendation-based editor for business process modeling." In: *Data & Knowledge Engineering* 70.6, pp. 483–503. DOI: 10.1016/j.datak.2011.02.002.

Krause, F., Bewernik, M.-A., and Fridgen, G. (2013). "Valuation of Manual and Automated Process Redesign from a Business Perspective." In: *Business Process Management Journal* 19.1.

La Rosa, M., Dumas, M., ter Hofstede, A. H. M., and Mendling, J. (2011a). "Configurable multi-perspective business process models." In: *Information Systems* 36.2, pp. 313–340. ISSN: 0306-4379.

La Rosa, M., Reijers, H. A., van der Aalst, W. M. P., Dijkman, R. M., Mendling, J., Dumas, M., and García-Bañuelos, L. (2011b). "APROMORE: An advanced process model repository." In: *Expert Systems with Applications* 38.6, pp. 7029–7040. ISSN: 09574174. DOI: 10.1016/j.eswa.2010.12.012.

La Rosa, M., ter Hofstede, A. H. M., Wohed, P., Reijers, H. A., Mendling, J., and van der Aalst, W. M. P. (2011c). "Managing Process Model Complexity via Concrete Syntax Modifications." In: *IEEE Transactions on Industrial Informatics* 7.2, pp. 255–265. DOI: 10.1109/TII.2011.2124467.

Laue, R. and Mendling, J. (2010). "Structuredness and its significance for correctness of process models." In: *Information Systems and e-Business Management* 8.3, pp. 287–307. ISSN: 1617-9846. DOI: 10.1007/s10257-009-0120-x.

Lautenbacher, F., Eisenbarth, T., and Bauer, B. (2009). "Process model adaptation using semantic technologies." In: *2009 13th Enterprise Distributed Object Computing Conference Workshops, EDOCW*, pp. 301–309. DOI: 10.1109/EDOCW.2009.5331985.

Mendling, J., Neumann, G., and van der Aalst, W. M. P. (2007a). "Understanding the Occurrence of Errors in Process Models Based on Metrics." In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, and Z. Tari. Vol. 4803. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 113–130. ISBN: 978-3-540-76846-3. DOI: 10.1007/978-3-540-76848-7_9.

Mendling, J., Reijers, H. A., and Cardoso, J. (2007b). "What Makes Process Models Understandable?" In: *Business Process Management*. Ed. by G. Alonso, P. Dadam, and M. Rosemann. Vol. 4714. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–63. ISBN: 978-3-540-75182-3. DOI: 10.1007/978-3-540-75183-0_4.

Mendling, J., Reijers, H. A., and van der Aalst, W. M. P. (2010). "Seven process modeling guidelines (7PMG)." In: *Information and Software Technology* 52.2, pp. 127–136. ISSN: 0950-5849.

Minor, M., Tartakovski, A., and Bergmann, R. (2007). "Representation and Structure-Based Similarity Assessment for Agile Workflows." In: *Case-Based Reasoning Research and Development*. Ed. by R. O. Weber and M. M. Richter. Vol. 4626. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 224–238. ISBN: 978-3-540-74138-1. DOI: 10.1007/978-3-540-74141-1_16.

Montani, S., Leonardi, G., Quaglini, S., Cavallini, A., and Micieli, G. (2015). "A knowledge-intensive approach to process similarity calculation." In: *Expert Systems with Applications* 42.9, pp. 4207–4215. ISSN: 09574174. DOI: 10.1016/j.eswa.2015.01.027.

Moreno-Montes de Oca, I., Snoeck, M., Reijers, H. A., and Rodríguez-Morffi, A. (2015). "A systematic literature review of studies on business process modeling quality." In: *Information and Software Technology* 58, pp. 187–205. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2014.07.011.

Munoz-Gama, J., Carmona, J., and van der Aalst, W. M. (2014). "Single-Entry Single-Exit decomposed conformance checking." In: *Information Systems* 46, pp. 102–122. ISSN: 0306-4379. DOI: 10.1016/j.is.2014.04.003.

Polyvyanyy, A., Vanhatalo, J., and Völzer, H. (2011). "Simplified Computation and Generalization of the Refined Process Structure Tree." In: *Proceedings of the 7th International Conference on Web Services and Formal Methods*. WS-FM'10. Berlin, Heidelberg:

Springer-Verlag, pp. 25–41. ISBN: 978-3-642-19588-4. URL: http://dl.acm.org/citation.cfm?id=1987781.1987783.

Russell, N., ter Hofstede, A. H. M., and Mulyar, N. (2006a). "Workflow ControlFlow Patterns: A Revised View." In: *BPM Center Report BPM-06-22*. URL: http://bpmcenter.org/reports.

Sánchez González, L., García Rubio, F., Ruiz González, F., and Piattini Velthuis, M. (2010). "Measurement in business processes: A systematic review." In: *Business Process Management Journal* 16.1, pp. 114–134. ISSN: 1463-7154. DOI: 10.1108/14637151011017976.

Van der Aalst, W. M. P. (2013). "Business process management: A comprehensive survey." In: *ISRN Software Engineering* 2013.

Van der Aalst, W. M. P., Adriansyah, A., de Medeiros, A., Arcieri, F., Baier, T., Blickle, T., Bose, J., van den Brand, P., Brandtjen, R., and Buijs, J. (2012). "Process Mining Manifesto." In: *BPM 2011 Workshops, Part I, LNBIP 99*, pp. 169–194.

Van der Aalst, W. M. P., Mylopoulos, J., Sadeh, N. M., Shaw, M. J., Szyperski, C., and Mendling, J. (2008). *Metrics for Process Models*. Vol. 6. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-540-89223-6. DOI: 10.1007/978-3-540-89224-3.

Van der Aalst, W. M. P., Rubin, V., Verbeek, H. M., van Dongen, B. F., Kindler, E., and Günther, C. W. (2010). "Process mining: a two-step approach to balance between underfitting and overfitting." In: *Software & Systems Modeling* 9.1, pp. 87–111. ISSN: 1619-1366.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Van der Aalst, W. M. P., Weijters, A. J. M. M., and Maruster, L. (2004). "Workflow mining: Discovering process models from event logs." In: *IEEE Transactions on Knowledge and Data Engineering* 16.9, pp. 1128–1142.

Vanhatalo, J., Völzer, H., and Koehler, J. (2008a). "The Refined Process Structure Tree." In: *Business Process Management*. Ed. by M. Dumas, M. Reichert, and M.-C. Shan. Vol. 5240. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 100–115. ISBN: 978-3-540-85757-0. DOI: 10.1007/978-3-540-85758-7_10.

Vanhatalo, J., Völzer, H., and Koehler, J. (2009). "The refined process structure tree." In: *Data & Knowledge Engineering* 68.9, pp. 793–818. DOI: 10.1016/j.datak.2009.02.015.

Vanhatalo, J., Völzer, H., Leymann, F., and Moser, S. (2008b). "Automatic Workflow Graph Refactoring and Completion: Service-Oriented Computing – ICSOC 2008." In: *Service-Oriented Computing – ICSOC 2008*. Ed. by A. Bouguettaya, I. Krueger, and T. Margaria. Vol. 5364. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 100–115. ISBN: 978-3-540-89647-0. DOI: 10.1007/978-3-540-89652-4_11.

Verbeek, H. M. W., Pretorius, A. J., van der Aalst, W. M. P., and van Wijk, J. J. (2007). "Visualizing state spaces with Petri nets." In: *Computer Science Report* 7.01.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Wetzstein, B., Ma, Z., Filipowska, A., Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., and Cicurel, L. (2007). "Semantic Business Process Management: A Lifecycle Based Requirements Analysis." In: *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007) in conjunction with the 3rd European Semantic Web Conference (ESWC 2007)*.

2.2 Paper 2: Automated Planning of context-aware Process Models

Status	Published	Full Citation
Accepted	05/2015	Heinrich, B., Schön, D. (2015), Automated Planning of context-aware Process Models, Proceedings of the 23rd European Conference on Information Systems (ECIS 2015), May 26-29, Münster, Germany.

Abstract

Most real world processes are heavily influenced by environmental factors, which are referred to as the context of a process. Thus, the consideration of context is proposed within the research field of Business Process Modeling. Most existing context-aware modeling approaches consider context only in terms of static information like, for instance, the location where a process is performed. However, context information like the weather could change during the conduction of a process, which we will denote as non-static context. In order to increase the flexibility concerning environmental influences in general and especially context-related events of context-aware processes, we present an approach for the automated planning of context-aware process models that considers static and non-static context. We therefore propose an extended state transition system in order to represent context information in terms of context variables and consider process exogenous changes of these context variables through context signals and receive context actions. Further, to ensure a correct, complete and time efficient construction of context-aware process models, a planning approach is used to support modelers by means of an algorithm. To demonstrate the feasibility of our approach we mathematically evaluated the algorithm and applied it to real world processes.

Keywords: Context Awareness, Business Process Modeling, Automated Planning

Post publication changes:

- Several formatting changes for consistency reasons
- Changed citation style for consistency reasons
- Fixed usage of commata in combination with “e.g.” and “cf.” in citations
- The numbering of headlines, tables, figures, definitions and algorithms was changed for consistency reasons
- Consistent capitalization of Section references

2.2.1 Introduction

Most real world processes are heavily influenced by environmental factors such as context-specific information (Hu et al., 2012; Soffer, 2005; Wang et al., 2012; Zhou et al., 2011). Hence, to appropriately react on such influences, processes have to be context-aware (Gottschalk et al., 2010; Gottschalk et al., 2007; La Rosa et al., 2011a; Reichert et al., 2012; Swenson, 2010; van der Aalst et al., 2006). As Abowd et al. (1999) and Dey (2001) define context as "any information that can be used to characterize the situation of an entity" and "an entity is a person, place, or object that is considered relevant" within a process, we call processes that are adaptive to environmental influences context-aware. Especially emerging modern technologies like smartphones, wearable sensors and mobile devices in general (Baldauf et al., 2007; Zhang et al., 2009) enable to gather detailed information about the current context of an entity and especially of the performer of a process. Context information like the current location, for instance, could easily be gathered by using sensor technology, which is present in almost every modern smartphone. Thus, the research field of context-aware processes of both, organizations and private persons, becomes even more relevant especially with respect to mobile processes and pervasive computing (Bettini et al., 2010; Satyanarayanan, 2001; Ye et al., 2012). As process models are an established method to represent processes, the consideration of context-specific information in process models in order to increase the flexibility of context-aware real world processes is also discussed (e.g., Schonenberg et al., 2008).

To address context-aware process models, several approaches (cf., e.g., Bucchiarone et al., 2013; Hallerbach et al., 2008a; Hallerbach et al., 2010; Rosemann et al., 2010; Rosemann et al., 2007; Tealeb et al., 2014; van der Aalst et al., 2006) have been presented in the last years. All of those approaches consider context by means of appropriate configurations or variants of a so called master process model or segments of a process model. Each of these configurations or variants is created before starting the process execution (i.e., at design time) and is valid in a specific context. Then, these approaches select an appropriate configuration or variant from a process repository based on the current context during the execution of the process (i.e., at runtime). A few other authors follow a different approach (cf., e.g., Ayora et al., 2013; Sakurai et al., 2012) and consider context information only during runtime of a process. They do not consider any context information at design time and within the process model. The (partial) consideration of context during runtime usually requires the performer of the process to provide a broad knowledge and a high level of expertise, which leads to restrictions especially regarding complex processes. Moreover, it could result in an increased process execution time as the consideration of context information during runtime costs time (cf., e.g., Fujii et al., 2009).

Summing up, current approaches have already strongly contributed to the current knowledge about context-aware processes. But most of them consider context in terms of different but static environments like different locations in which a process can be performed. However, occurring environmental events (i.e., context events) could change the context of a process *throughout its conduction*, which we will denote as "*non-static context*". A simple, yet common example is an upcoming thunderstorm during the conduction of an outdoor process. Here, a process that has already started to be conducted or a selected process variant may become inadequate due to a context event. For example, subsequent actions could no longer be performed (e.g., a flight could not depart due to a storm) or already running actions might be terminated (e.g., an emergency landing is required because of a technical issue). In the worst case, the conduction of the complete process must be terminated, as the desired process goal could no longer be reached. Therefore, it is inevitable to consider *non-static context* throughout a process model (Dobson et al., 2006). This issue is addressed only by few authors at all, which, however, use rather complex process reconfiguration tasks that need to take place during runtime (cf., e.g., Hallerbach et al., 2008a) and therefore may delay the process execution. Thus, the consideration already during design time and hence the reduction of complexity during runtime is useful. However, to the best of our knowledge, there is no approach, which considers *non-static context* during design time within process models so that no further, complex reconfiguration tasks need to be performed during runtime to enable a consideration of context. We therefore want to address this research question and tend to construct a *comprehensive* process model, which does not have to be reconfigured or adapted during runtime in order to consider *non-static context*.

To allow us a correct, complete and fast construction of comprehensive context-aware process models the second research question arises of how this construction can be supported through a planning approach (e.g., algorithms). This question follows several approaches to support modelers and business analysts by means of automation (e.g., algorithms) in the last years. For instance, Process Mining (e.g., van der Aalst et al., 2004), especially automated process discovery, assists business analysts in the *process analysis* phase of the Business Process Management (BPM) Lifecycle (cf. Wetzstein et al., 2007). Automated service selection and composition (cf., e.g., Heinrich et al., 2015a; Khan et al., 2010; Weber, 2007) increase the degree of automation within the phases *process implementation* and *process execution*. The construction of process models in an automated way is addressed by process planning algorithms (Heinrich et al., 2012; Henneberger et al., 2008; Hoffmann et al., 2009; Hoffmann et al., 2012) to support modelers in the phase of *process modeling*. To follow such approaches, we aim to present an approach for the *automated planning* of context-aware process models (second research question)

that can cope with *non-static context* (first research question) in this paper. The main contributions are as follows:

❶ We aim to construct comprehensive context-aware process models by considering *non-static* context in terms of context variables. Therefore, we introduce a formal definition of a *planning domain* that can cope with *non-static context* throughout a process model. To construct complex process models, which are able to cope with occurring environmental events in terms of context events (resulting in context signals), we take into account that the context of the process might change while it is performed later on.

❷ To address the issue of constructing context-aware process models in an automated way, we present a novel *planning algorithm*. In order to abstract from an individual process execution (cf. Ghallab et al., 2004) and to construct entire process models, we consider belief states instead of world states. Further, to enable a widespread use and acceptance of our approach we present a nondeterministic state transition system that is independent from a specific process modeling language as the formal basis for our algorithm.

2.2.2 Background

In the following, we will discuss existing approaches on context-aware BPM and related topics like modeling exceptional flows and signals. Further, we want to summarize related work within the fields of context representation in general and well-known process modeling languages in particular.

An increasing amount of research has been conducted on context-aware BPM. Rosemann et al. (2008) state that context-awareness is highly relevant in the field of business process modeling. Rosemann et al. (2010) further state that the research has "increased attention to flexibility" which results from "the trend toward decreasing time-to-market" and "increasing frequency of product innovation" and therefore a demand for process flexibility with regard to their environment (Soffer, 2005) emerges. Rosemann et al. understand process flexibility as "the ability to change the process [during runtime] without completely replacing it" (Bider, 2005; Regev et al., 2007). Hallerbach et al. (2010) follow a slightly different approach by defining so called process variants. By means of process variants, they assign different process models to different contexts. Further, during runtime, they reconfigure the process for the current context by applying rather complex reconfiguration tasks, defined alongside with the process variants. Basically, their approach could be understood as a selection of a specific process model from a large repository of regular (not context-aware) process models and a reconfig-

uration of the currently conducted process model in case of changing context information. As both approaches need to change parts of the process during runtime they do not consider a comprehensive process model that takes context into account as part of the regular control flow, which would be favorable with respect to an automated process execution (cf., e.g., Khan et al., 2010; Weber, 2007).

Zhu et al. (2014a) and Zhu et al. (2014b) strive the problem of location dependency, which is a subproblem of context-awareness, by means of "location-dependent process model patterns" (e.g., a location dependent exclusive choice). However, these location-dependent process model patterns do not significantly differ from regular control flow patterns (van der Aalst et al., 2003). For example, a "location-dependent exclusive split" (i.e., an exclusive choice) describes a decision based on location information and hence is basically a particular type of a regular exclusive choice. Further, they do not address other context information and the challenge of non-static context (Dobson et al., 2006).

Mattos et al. (2014) propose a metamodel for context-aware process models but do not address how to construct and represent (in terms of a notation) context-aware process models. Further, they state that well-known modeling languages like EPC, UML or BPMN "do not include the concept of context" but also leave this issue unsolved. Within their metamodel they define *Contextual Entities* (persons, places, objects, etc.) which are considered in terms of *Rules* that must be met in order to execute an action but they do not cope with the problem of integrating context into process models as they do not propose an approach to represent context and consider it in terms of a modeling language.

However, there exist well-known approaches striving related challenges like exception and signal handling in process models. Russell et al. (2006b) present five different exception types that could be determined throughout a process. Amongst others, *External Triggers* represent "[t]riggers from sources external to a work item [that] are often used as a means of signaling the occurrence of an event that impacts on the work item and requires some form of handling. These triggers are typically initiated [...] from processes in the operational environment, in which the PAIS [i.e., Process Aware Information System] resides" (Russell et al., 2006b). Thus, environmental events in terms of context signals could be classified as *External Triggers*. Here, Russell et al. (2006b) focus on recovery and termination strategies in order to enable a graceful continuation or termination of the process. For example, they consider rollback strategies and reoffering the interrupted action later. Thus, as we want to design process models that are adaptive to changing contexts, their findings are not applicable for our needs.

Besides the consideration of exception handling in terms of recovery approaches, UML and BPMN contain approaches to denote exceptional flows, which may also be used to represent context signals within process models. BPMN (since version 2.0)

distinguishes between erroneous situations that lead to an interruption of an action (*error events*) and so called *escalation events* that may be non-interrupting (*Business Process Model and Notation (BPMN): Version 2.0.2* 2013). Context signals may influence an action in a non-interrupting and an interrupting manner, too, and thus both cases should be considered here, as well. Furthermore, UML (cf. *OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015) contains so called *Exception Handlers* that, in case of an exception, gracefully handle the exception that occurred during a performed action (i.e., interrupting). Further, so-called *Interruptible Activity Regions* with *interruptible edges* denote areas in a process model (instead of single actions) that could be interrupted due to an exception. Besides, exogenous influences (i.e., non-interrupting) are denoted in terms of *signals*. Incoming signals (e.g., events initiated by another process) are handled by so called *Accept Event Actions* that initiate a further regular control flow within the process. "If [an] accept event action is executed and object detected event matching one of the triggers [i.e., signals] on the action, then the accept event action outputs a value describing the event. If the event does not match expected event, the action waits for the next event." In contrast to the representation of exceptions, events and signals, the representation of specific *context variables* and their consideration in process models receives only little discussion so far (cf. Zhu et al., 2014a). Process modeling languages like BPMN, UML or EPC do not take context variables explicitly into account. In BPMN, for example, the context could be modeled by means of swimlanes, annotations or data (*Business Process Model and Notation (BPMN): Version 2.0.2* 2013). Such a consideration in annotations or data attributes is limiting, as context then could not be considered as a variable during process execution (cf., e.g., Mulholland et al., 2006) or by means of the control flow (e.g., decisions in terms of an exclusive choice are not feasible on annotations). Further, in languages like BPMN and UML (cf., e.g., *Business Process Model and Notation (BPMN): Version 2.0.2* 2013; *OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015) no modeling elements for context signals or context variables in particular exist, even though they strive related issues. To summarize, the discussed languages provide a basis to cope with contribution ❶ as they address related issues. However they do not strive contribution ❷, the automated construction of context-aware process models.

As modeling context information is the foundation for our work, we will review current approaches for *context representation* in the following as well. Strang et al. (2004) and Bettini et al. (2010) reviewed the most relevant approaches (according their appraisal) for modeling context and classified them in several groups. Especially the identified Ontology Based Models are closely related to the research strand of process planning (cf. Section 2.2.3). Further, ontology based context modeling has been proposed due to its advantages by means of normalization and formality (Öztürk et al., 1997). We will

therefore use ontology based context modeling as a further basis to cope with contribution ❶ within our approach. Basically all approaches of this group (cf., e.g., Attard et al., 2013; Chen et al., 2003; Nadoveza et al., 2014; Simons et al., 2007) have in common, that context is represented as a set of atomic or composite (consisting of atomic or composite variables again) context variables with a specific domain. Simons et al. (2007), for example, introduce a UML profile for context modeling that could be used as a comprehensive metamodel for context modeling. Chen et al. (2003) and Wang et al. (2004) introduce context ontologies expressed in the Web Ontology Language (OWL), which is heavily used within the research fields of BPM and automated compositions of (web) services. However, none of these works addresses the issue of planning context-aware process models and thus contribution ❷, as they focus on the representation of context.

2.2.3 Planning Domain and Running Example

As already stated, several approaches to support modelers and business analysts by means of process automation have been proposed in the last years. As part of this development, the construction of context-aware process models in an automated way seems promising especially as context-aware process models are usually more complex than non-context-aware process models. The automated construction of process models can be understood as a planning problem (e.g., Heinrich et al., 2009). More precisely, we have to abstract from an individual process execution and its world states in order to construct entire process models, valid for various process executions, resulting in a non-deterministic planning problem with belief states (Ghallab et al., 2004). Here, a belief state represents possibly infinite sets of world states. Hence, to enable a widespread use and acceptance of our approach to construct context-aware process models, we use a general set-theoretic planning domain (cf. Ghallab et al., 2004) as a foundation and starting point for our approach. This further guarantees a maximum of compatibility with existing approaches in the literature (e.g., Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2009; Meyer et al., 2006; Sycara et al., 2003). Considering this planning domain, a *planning graph,* which basically consists of two types of nodes - representing *belief states* and *actions* - and edges is used. As a belief state $bs \subseteq BST$ could be seen as a set of information about the variables currently available in a process state (so called belief state variables) we denote a belief state as a set of *belief state tuples*, whereas each of them denotes one particular characteristic:

Definition 2.2.1. (*belief state*). BS is a finite set of belief states. A belief state $bs \in BS$ contains a set BST of belief state tuples. Here, every belief state tuple p exists one time at the most. A belief state tuple $p \in BST$ is a tuple of a belief state variable $v(p)$ and a subset $r(p)$ of its predefined domain $dom(p)$, which we will write as $p := (v(p), r(p))$.

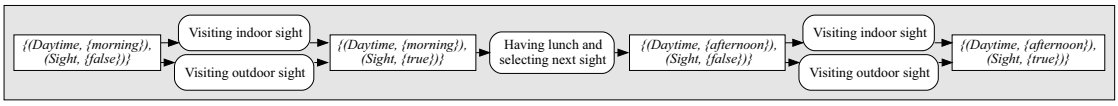

Figure 2.5: Initial Planning Graph of the Running Example

To represent *actions* conducted by a so called performer, a second type of node is defined. An action a is a triple $a{=}(name(a), pre(a), \mathit{eff}(a))$ whereas $pre(a){\subseteq}BST$ denotes all conditions when a could be applied and $\mathit{eff}(a) \subseteq BST$ denotes all changes that result from the conduction of a.

Definition 2.2.2. (*action*). A is a finite set of actions. Each action $a{\in}A$ is a triple consisting of the action name and two sets, which we will write as $a{:=}(name(a), pre(a), \mathit{eff}(a))$. The set $pre(a){\subseteq}\ BST$ are the preconditions of a and the set $\mathit{eff}(a){\subseteq}\ BST$ are the effects of a.

In a next step, we define a planning graph as follows:

Definition 2.2.3. (*planning graph*). A planning graph is an acyclic, bipartite, directed graph $G{=}(N, E)$, with the set of nodes N and the set of edges E. Henceforth, the set of nodes N consists of two partitions: First, the set of action nodes $Part_{\mathrm{A}}$ (set A of actions) and second the set of belief state nodes $Part_{\mathrm{BS}}$ (set BS of belief states). Each node $bs{\in}Part_{\mathrm{BS}}$ is representing one distinct belief state in the planning graph. The planning graph starts with one initial belief state and ends with one to probably many goal belief states (with $Init{\in}BS$ and $Goal_{\mathrm{j}}{\in}BS$).

To illustrate our approach and the planning domain, we will use a simplified excerpt of a real world tourism process, which guides a tourist during a day trip to a foreign city and is highly influenced by context information. The regular, non-context-aware process is represented by the planning graph shown in Figure 2.5. Here, the initial belief state (leftmost belief state; belief states denoted as rectangles with italic text) represents the start of a tourist's day trip in the morning. Thereby, in the morning the tourist could perform the actions *Visiting indoor sight* or *Visiting outdoor sight* (actions denoted as rounded rectangles with regular text). Afterwards s/he selects the next sight to visit while having lunch and visits an outdoor or indoor sight in the afternoon again. After performing one of those two actions, s/he finishes the process by reaching the goal belief state (rightmost belief state).

2.2.4 Approach to plan context-aware Process Models

To enable the automated planning of context-aware process models (cf. contributions ❶ and ❷ discussed in the introduction) we divide this overall goal into the following sub goals:

- *Set theoretic representation of context.* To consider context information in our planning domain, we need to adapt the above definitions. In particular, we have to extend the definition of belief states to denote context variables in terms of a set theoretic representation.

- *Consider context as non-static.* As context should be considered as non-static throughout a process (Dobson et al., 2006), it is required to take occurring process exogenous changes of context into account. Therefore, it is needed to extend the set theoretic state transition system by means of context signals that accord to those occurring process exogenous changes. In detail, to address contribution ❷, we need to automatically construct receive context actions (i.e., by means of an algorithm) that represent process exogenous changes of context (i.e., the transition) taking place during the conduction of a (regular) action in the process models.

- *Align notation to well-known modeling languages.* To ensure a maximum of compatibility with existing process modeling approaches and to increase the acceptance of the constructed process models among modelers, we need to consider existing notations of well-known process modeling languages like UML. Thus, in order to represent process exogenous context changes we need to use well-known modeling notation elements for asynchronous events or signals, for instance.

2.2.4.1 Consider Context within the Planning Domain

As a first step, we need to represent context variables in terms of our planning domain. As said above, context is supposed to be represented by means of ontology based models (Bettini et al., 2010; Öztürk et al., 1997; Strang et al., 2004). Thus, we represent the context of a process as a set of so called context variables, a subset of belief state variables that allow us to denote context information and their predefined domains. As context variables represent process exogenous information and regular belief state variables represent process endogenous information, we define those two set as disjoint. A belief state tuple $p \in BST$ therefore is either contained in the set $CT \subseteq BST$ of so called context tuples or in the set $BST \setminus CT$ of regular belief state tuples.

Therefore, we adapt the previous definitions of belief states and actions as follows:

Definition 2.2.4. (*belief state*). BS is a finite set of belief states. A belief state $bs \in BS$ contains a set BST of belief state tuples. A belief state tuple $p \in BST$ is either assigned to the set of context tuples $CT \subseteq BST$ or to the set of regular belief state tuples $BST \setminus CT$ and is, in any case, a tuple of a belief state variable $v(p)$ and a subset $r(p)$ of its predefined domain $dom(p)$, which we will write as $p{:=}(v(p),r(p))$.

Definition 2.2.5. (*action*). A is a finite set of actions. Each action $a \in A$ is a triple consisting of the action name and two sets, which we will write as $a{:=}(name(a), pre(a), \mathit{eff}(a))$. The set $pre(a) \subseteq BST$ are the preconditions that must be met to apply action a. The set $\mathit{eff}(a) \subseteq BST \setminus CT$ are the effects that result from the conduction of a.

By using context tuples in the preconditions of actions, we create a basis for considering context information throughout a process model. However as Definition 2.2.5 also outlines, the conduction of (regular) actions a cannot change context variables due to their process exogenous character.

Both definitions are a first step that allows us to take context into account in a static manner. For instance, within our running example, it may be favorable to plan the actions *Visiting indoor sight* and *Visiting outdoor sight* depending on the weather, as a performer may prefer an indoor activity like a museum to an outdoor activity in case of bad weather. Such user preferences can be denoted and considered by means of belief state tuples within the initial state of the process model. To denote the fact that visiting an outdoor sight depends on the weather, we include the according context tuple (*Rain*, {*false*}) in the preconditions of the corresponding action. By denoting context information in terms of context variables we are now able to consider static context information throughout the process.

In order to take different possible contexts in which a planned process model can be performed into account, we need to identify relevant context variables that could influence the process execution. As each influencing context variable is part of the preconditions of at least one action, the identification is quite straightforward as we need to consider all context variables that are contained in any action's preconditions. Thus, in order to be able to determine if an action can be planned (i.e., is applicable) within a particular context, that is, if all context-related preconditions of that action hold, context variables must be included from the beginning to the end of the regular process, making this process context-aware in a static manner. Moreover, it is needed to include all relevant context variables (i.e., all context variables present in any actions preconditions) within the initial belief state of the process, as these context variables cannot be changed by regular actions (cf. Definition 2.2.5).

A comprehensive coverage of possible contexts, in which a process can be performed, requires the consideration of all context variables, contained in the preconditions of any action $a \in A$. Thereby, all possible contexts can be represented in terms of permutations of context variables. Therefore we include the following context tuples $(v(p), r(p)) \in CT$ to the initial states of the state transition system:

$\{(v(p), r(p)) \in CT \mid r(p) \subseteq dom(p), \exists (v(q), r(q)) \in pre(a), v(q) = v(p), a \in A, q \in CT\}$.

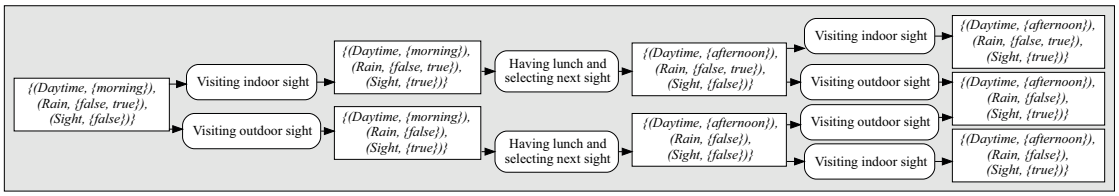

Figure 2.6: Extended Planning Graph including Static Context Information

By means of this extension, our running example can be extended as seen in Figure 2.6.

2.2.4.2 Consider non-static Context within the Planning of Process Models

So far, context is considered as static throughout the process, whereas in real world processes changes to context variables could occur when conducting the process due to environmental influences (Dobson et al., 2006). For example, it could start raining while conducting the action *Visiting outdoor sight* and thus, the performer may be required to react on this dynamic context change. Thus, we have to address how to cope with non-static context throughout the process in the following.

To consider process exogenous changes of the values of context variables caused by nature or process-external agents but without the involvement of a performer of the process, we need to model this exogenously initiated state transition. We denote a specific event, which may occur dynamically throughout the process as a *context signal*. As a context signal represents an occurred event or state, we denote a context signal $sig \in SIG$ (with SIG as the set of context signals) as a set of context-related belief state tuples $sig{:=}\{(v(p),r(p))|(v(p),r(p))\in CT\}$. For example, the context signal *rain starts falling*:= $\{(Rain, \{true\})\}$ denotes, that the value of the context variable *Rain* changed to *true* due to a process exogenous event. We define a context signal as follows:

Definition 2.2.6. (*context signal*). *SIG* is a finite set of context signals. A context signal $sig{\in}SIG$ contains a set of context tuples with $(v(p),r(p))\in sig\subseteq CT$.

Besides the state representation, we have to cope with the state transition, which means, we need a model element that depicts the according transition of context variables (i.e., the value of the context variable *Rain* changes from *false* to *true*). Thus, we define a *receive context action* $c_{\text{sig}} \in C$ (with C as the set of receive context actions) similar to a regular action (in terms of preconditions and effects) but distinct the sets of receive context actions and regular actions. This is due to the fact that regular actions need to be executed by a performer and could not influence context variables while receive context

actions are triggered by nature or (process-external) agents who are not conducting the process. Thus, we define a receive context action as follows:

Definition 2.2.7. (*receive context action*). C is a finite set of receive context actions. A receive context action $c_{\text{sig}} \in C$ – related to a context signal *sig* – is defined as a triple consisting of the name of the context signal that is handled by this receive context action and two sets, which we will write as $c_{\text{sig}} := (name(c_{\text{sig}}), pre(c_{\text{sig}}), \mathit{eff}(c_{\text{sig}}))$. The first set $pre(c_{\text{sig}}) \subseteq BST$ are the preconditions of c_{sig} and the second set $\mathit{eff}(c_{\text{sig}}) = sig \subseteq CT$ are the effects of c_{sig}.

Context signals are process exogenous and thus independent of regular belief state variables representing information created within the process. Therefore, within the effects of receive context actions only context variables are considered. However, as process endogenous information may be used to determine if a receive context action is applicable, regular belief state variables may be used within the preconditions of receive context actions. Within our running example the receive context action for the context signal *rain starts falling*, for instance, needs to be considered only if the performer selects an outdoor sight (denoted by the belief state variable *NextSight*) and we therefore include (*NextSight*, {*outd.*}) in the preconditions of the according receive context action.

Considering our planning domain, the determination whether an action is applicable in a belief state is based on the according preconditions and the belief state itself. If the preconditions of an action hold in a belief state, the action is applicable, if the preconditions do not hold, the action is not applicable. Based on this, context signals that may influence the execution of an action are identified by analyzing the preconditions of this action.

Following this, an action would be interrupted if the preconditions of a (regular) planned action no longer hold (due to an occurred context signal). For example, within our running example the action *Visiting outdoor sight* will be interrupted by the mentioned context signal

$$\textit{rain starts falling}\ (\mathit{eff}(c_{\text{rain starts falling}}) = \{(Rain, \{true\})\})$$

as

$$(Rain, \{false\}) \in pre(\textit{Visiting outdoor sight}) \text{ and } true \notin \{false\}$$

which is the intuitive way to denote that the planned action must not be performed in case of bad weather. Thus, formally written, a planned action a must be interrupted by a context signal *sig* if and only if

$$\exists\ (v(p), r(p)) \in sig : \exists\ (v(q), r(q)) \in pre(a), v(q) = v(p), r(p) \cap r(q) = \varnothing.$$

Furthermore, context signals may also be relevant for actions that are not necessarily interrupted by the context signal. For example the context signal *rain stops falling* may occur during *Visiting indoor sight* and thus a performer may continue this action as sunny weather does not prevent him from visiting an indoor sight while another performer might prefer *Visiting outdoor sight* to *Visiting indoor sight* if the rain stops falling. This means that such a context signal may lead to an interruption under certain circumstances but does not necessarily interrupt the considered action. Hence, the further process is conditionally influenced by the context signal. In terms of our planning domain, this means, that the restrictions of at least one context tuple of the considered context signal overlap with those of the preconditions of the planned action. Regarding our example, $r(Rain) = dom(Rain)$ is present in the preconditions of the action *Visiting indoor sight* and thus, the action is applicable regardless of the weather. However, it may be interrupted if a performer prefers outdoor activities. Based on such user preferences it can be decided whether to continue or to interrupt the conduction of the action based on its preconditions. In terms of our planning domain, we distinct both cases due to the fact, that the planned action is applicable again in the belief state after the occurrence of a context signal.

Based on the Definitions 2.2.4 to 2.2.7, we are now able to define a nondeterministic context-aware state transition system as follows:

Definition 2.2.8. (*Nondeterministic context-aware state transition system*). A nondeterministic context-aware belief state transition system is a tuple $\Sigma = (BS, A, C, R)$, where

- BS is a finite set of belief states. A belief state $bs \in BS$ contains a set BST of belief state tuples. Here every belief state tuple p exists one time at the most. A belief state tuple $p \in BST$ is a tuple of a belief state variable $v(p)$ and a subset $r(p)$ of its predefined domain $dom(p)$, which we will write as $p{:=}(v(p),r(p))$. The set of context tuples is denoted as $CT \subseteq BST$.

- A is a finite set of actions. Each action $a \in A$ is a triple consisting of the action name and two sets, which we will write as $a{:=}(name(a), pre(a), eff(a))$. The set $pre(a) \subseteq BST$ are the preconditions of a and the set $eff(a) \subseteq BST \setminus CT$ are the effects of a.

- An action a is *applicable* in a belief state bs if $\forall u \in pre(a)\ \exists\ w \in bs:\ v(u){=}v(w) \wedge (r(u) \cap r(w) \neq \varnothing)$.

- SIG is a finite set of context signals. A context signal $sig \in SIG$ contains a set of context tuples with $(v(p),r(p)) \in sig \subseteq CT$. An according receive context action $c_{\text{sig}} \in C$ further is defined as a triple consisting of the name of the context signal handled

by this receive context action and two sets, which we will write as c_{sig}:=(*name*(c_{sig}), *pre*(c_{sig}), *eff*(c_{sig})). The set *pre*(c_{sig})$\subseteq$*BST* are the preconditions of c_{sig} and the set *eff*(c_{sig})=*sig*$\subseteq$*CT* are the effects of c_{sig}.

- The context signal *sig* that is handled by a receive context action c_{sig} interrupts a planned action *a* in a belief state *bs* if $\exists\, t \in eff(c_{sig}) : \exists\, u \in pre(a), v(t) = v(u) \wedge (\forall\, x \in pre(c_{sig}) \exists\, w \in bs : v(w) = v(x) \wedge ((\nexists\, y \in pre(a), v(y) = v(x), r(x) \cap r(w) \neq \varnothing) \vee (\exists\, y \in pre(a), v(y) = v(x), r(x) \cap r(y) \cap r(w) \neq \varnothing))$.
- *R: BS*×*A*×*C* → 2^{BS} is the transition function. *R* associates to each belief state *bs*$\in$*BS*, each action *a*$\in$*A* and each receive context action $c_{\text{sig}}\in C$ the set *R*(*bs*, *a*, c_{sig})$\subseteq$*BS* of next belief states.

The transition function *R* basically consists of two parts. In the first part, the belief state resulting from the application of an action *a* is retrieved. Further, in the second part, if a context signal *sig* interrupts the action *a*, the according belief state resulting from the interruption is determined taking into account the according receive context action c_{sig}.

More precisely, in the first part the belief state *bs* prior to *a* is joined with the preconditions of *a* while the effects of *a* are concatenated with *bs* by means of a set theoretic union of each belief state variable's restriction to retrieve the belief state resulting from the application of *a*. For example, the belief state variable *Sight* is changed to *true* by the action *Visiting outdoor sight* within our running example.

In the second part, the probable interruption by means of the context signal *sig* has to be taken into account. Here, it is uncertain (during design time) whether and to which extent the effects of the interrupted action *a* have to be applied. Thus, within the belief state after the interruption, the value of a belief state variable could either be the value that is defined in the effects of the action *a* (if the effects are applied) or the value of the variable in the belief state *bs* prior to *a*. Within our example, the variable *Sight*, as already stated, is changed to *true* in case the action *Visiting outdoor sight* is applied. As the value is *false* in the belief state prior to this action, we have to unify those two values so that *Sight* may be *false* or *true* in the next belief state after the interruption. This can be referred to as context-related uncertainty (corresponding to the well-known initial state uncertainty). However, when we determine the belief state after the interruption the particular context signal that interrupts the action has to be considered as well. This means, the preconditions and effects of the according receive context action have to be applied to the previously constructed belief state regarding our state transition function in Definition 2.2.8. Thus, considering the context signal *rain starts falling*, the value of the context variable *Rain* is *true*, which is the unique representation of the state, as the value

of a context variable that is addressed by a context signal is known (i.e., no context-related uncertainty in this case).

After both steps, if the resulting belief state already exists within the planning graph (e.g. in case a loop is created), the further planning for the current path stops here, as the belief state has already been checked for applicable actions.

By means of the nondeterministic context-aware state transition system in Definition 2.2.8, we are able to construct a comprehensive process model that considers context signals throughout the process. In order to visualize the resulting planning graph, we extend our running example by the two already mentioned context signals and their according receive context actions:

rain starts falling (
$pre(c_{\text{rain starts falling}})=\{(Rain, \{false\}), (NextSight, \{outd.\})\}$,
$eff(c_{\text{rain starts falling}})= \{(Rain, \{true\})\})$
rain stops falling (
$pre(c_{\text{rain stops falling}})=\{(Rain, \{true\})\}$,
$eff(c_{\text{rain stops falling}})= \{(Rain, \{false\})\})$

The context signal *rain starts falling*, for example, is relevant during the conduction of the action *Visiting outdoor sight* as the preconditions of this action will no longer hold in case of the occurrence of the context signal. Here, we additionally need the belief state variable *NextSight*, to avoid that the context signal *rain starts falling* interrupts the conduction of the action *Visiting indoor sight* and the according receive context action is planned, as no other action than *Visiting indoor sight* could be conducted in this case. As seen in Figure 2.7, receive context actions are represented in terms of Accept Event Actions (cf. *OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015) and thus depicted by means of concave pentagon symbols labelled with the name of the according context signals. Further, we denote context signals that are relevant for planned actions by means of Interruptible Activity Regions (cf. *OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015) and thus by means of dashed, light gray areas around these actions. Further, we connect an Interruptible Activity Region to the according receive context action that handles the context signal by means of a dashed edge.

In Figure 2.7 we see the process model as seen in Figure 2.6, denoted in terms of bold actions and bold belief states[2]. Based on Definition 2.2.8, for instance, during the conduction of the action *Visiting outdoor sight* (see leftmost area) the context signal *rain starts falling* may occur (see Interruptible Activity Region). If so, the regarding path continues with the according receive context action *rain starts falling* and continues with regular actions. Precisely, in the belief state right after the receive context action *rain*

[2]We deliberately omitted belief state tuples within belief states for reasons of readability. For a full version of Figure 2.7, including the belief state tuples see Appendix 7.2.1.

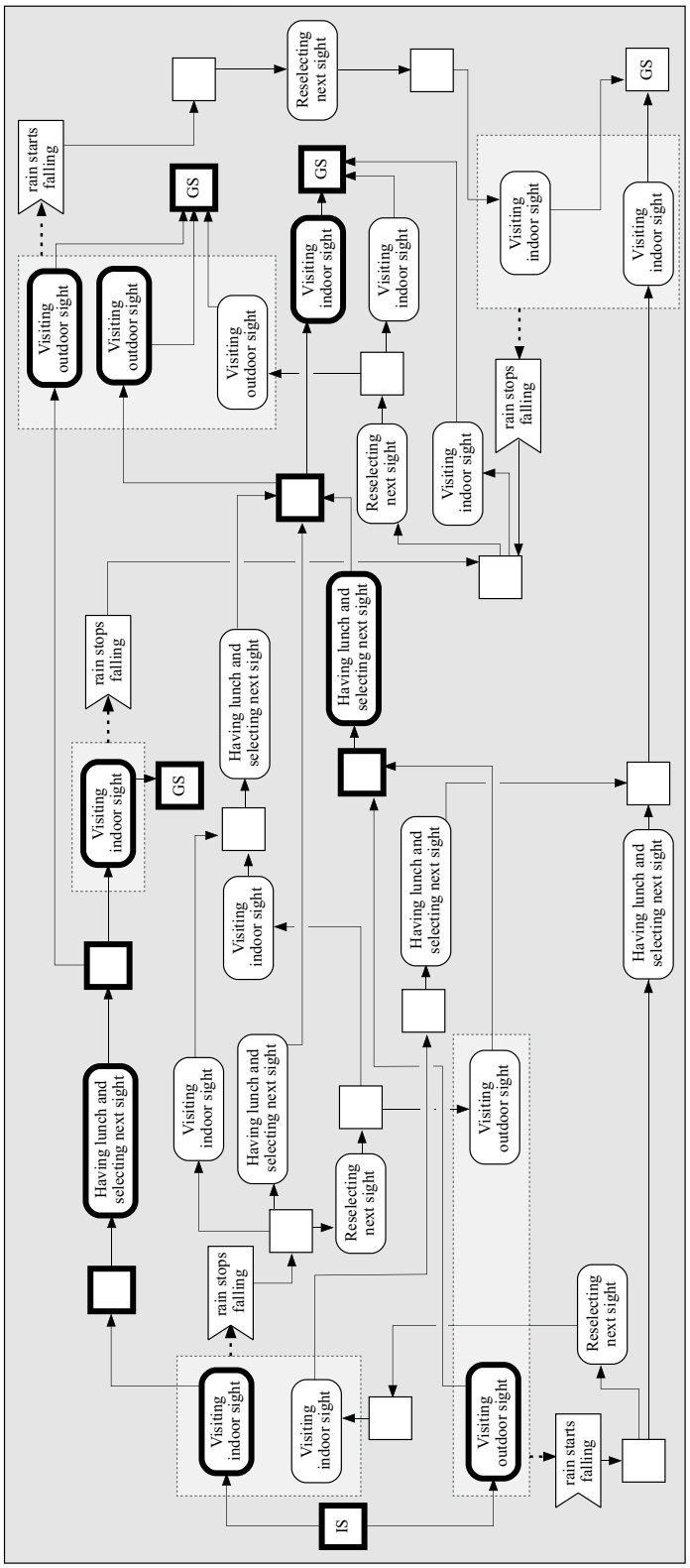

Figure 2.7: Planning Graph including Receive Context Actions and Interruptible Activity Regions

starts falling (see leftmost belief state), the actions *Having lunch and selecting next sight* and *Reselecting next sight* (as the user needs to choose a new sight) are applicable. Continuing this planning, the goal state (labeled with "GS") at the very right of Figure 2.7 can be reached.

Thus, by constructing a context-aware process model, we are able to consider possibly occurring context signals already during design time. We further enable to reduce complexity during runtime and thus allow performers to execute a process without being mindful of possible context signals.

2.2.4.3 Algorithm

Within an algorithm[3] that realizes the presented nondeterministic context-aware state transition system, the planning graph is constructed by means of a depth first search, starting with the initial belief state. Applicable actions are retrieved by means of their preconditions and thus, regarding the state transition function, the following belief states are constructed until a goal state or an already determined belief state is reached. Further, relevant context signals for all planned actions are retrieved (cf. Primitive `isRelevantForActionAndState`; **part of class ContextSignal**; Listing 2.1).

The retrieval basically consists of two components: First, a context signal that affects the conduction of a planned action is identified, if at least one context variable of the effects of the receive context action c_{sig} (i.e., of the according context signal *sig*) is also present in the preconditions of the action a (cf. $\exists t \in \mathit{eff}(c_{\text{sig}}) : \exists u \in \mathit{pre}(a),\ v(t)=v(u)$; lines 2-9). In a second step, it is checked if the context signal could occur (i.e., if all preconditions of the according receive context action c_{sig} are fulfilled within *bs*; cf. lines 12-20). If both conditions hold, c_{sig} is planned (cf. Definition 2.2.8).

Thereafter, the belief state after the planned receive context action c_{sig} is determined by means of the context-aware state transition function (cf. Primitive `applyContextSignal` **of class State**). Now, the algorithm continues planning applicable actions for this newly determined belief state regarding the transition function. The forward search stops if either no action is applicable in a belief state or if a goal state or an already considered belief state is reached.

2.2.5 Evaluation

We mathematically evaluated the feasibility of our approach by proving the key properties termination, completeness (i.e., all relevant receive context actions are considered in the resulting planning graph), correctness (i.e., no receive context actions are planned

[3] For the full pseudo-code of the algorithm, see Appendix 7.2.2.

Listing 2.1 Pseudo-Code for the retrieval of relevant context signals

```
1  boolean isRelevantForActionAndState(Action a, State previousState){
2    boolean foundCommonContextVariable = false;
3    for (ContextTuple signalTuple in effects) {
4      ContextTuple actionTuple = a.preconditions.find { e ~> e.variable ==
           signalTuple.variable};
5      if (actionTuple != null){
6        foundCommonContextVariable = true;
7      break;
8      }
9    }
10   if (!foundCommonContextVariable)
11     return false;
12   for (BeliefStateTuple signalTuple in preconditions) {
13     BeliefStateTuple actionTuple = a.preconditions.find { e ~> e.variable ==
           signalTuple.variable};
14     BeliefStateTuple stateTuple = state.bst.find { e ~> e.variable ==
           signalTuple.variable};
15     if (actionTuple != null && stateTuple != null){
16       BeliefStateTuple intersectionTuple =
             signalTuple.intersect(actionTuple).intersect(stateTuple);
17       if (intersectionTuple == null)
18         return false;
19     }
20   }
21   return true;
22 }
```

that must not interrupt a regular action) and computational complexity. Further, we evaluated our approach by means of a prototypical implementation of the previously presented algorithm and an experimental evaluation within real world fields of application.

2.2.5.1 Mathematical Evaluation

In order to prove the feasibility of our approach, we need to ensure that it terminates (Theorem 1), considers all receive context actions (completeness; Theorem 2) and does not plan any incorrect context signals and receive context actions (i.e., is correct; Theorem 3). It is proven, that the approach meets all these requirements and its computational complexity is $O(n^4)$ in the number of context tuples or belief state tuples, which means, the algorithm is computationally efficient (cf. Arora et al., 2009; Cobham, 1965). For the proof sketches see Appendix 7.2.3.

2.2.5.2 Prototypical Implementation and Experimental Evaluation

To demonstrate the practical feasibility (cf. Peffers et al., 2008) of our approach, we implemented the proposed algorithm by means of a prototype[4]. A Java implementation of a state transition system served as a basis for our work. We ensured the validity of our prototype by applying several tests using the JUnit framework including unit test, extreme value tests and integration tests. Further, persons other than the programmers validated the source code via a structured walkthrough.

In the next step, we applied our approach to the application fields *Day city trip* and *House building* to evaluate if the constructed context-aware process models are suitable for representing possibly occurring context signals and are thus valid (Adelman, 1991; Peffers et al., 2008)[5]. Here, our running example is a simplified excerpt of a Day city trip process that was defined based on the attractions and activity categories of trip planning websites like `tripadvisor.com`.

Further, to evaluate the efficacy of our approach, we conducted a naturalistic ex-post approach (cf. Venable et al., 2012) within the already mentioned application field *House building*. Within an experimental setting (cf. Hevner et al., 2004) we questioned, in a first step, an experienced architect, which context signals, from his experience, could possibly influence a House building process. This process, according to the architect, consists of three major phases that are required in order to build a house. After the planning

[4]We executed the prototype on a Dell Latitude Notebook with an Intel Core i7-2640M, 2.80 GHz, 8GB RAM running on Windows 8.1 (Version 6.3.9600) 64 bit and Java 1.8.0 (build 1.8.0.-b132) 64 bit.

[5]Thereby we used our prototypical implementation, which was also suitable for creating the figures within this paper.

phase, which is mainly not context-aware, the shell is constructed. This phase could be heavily influenced by environmental events like rain, raising or lowering temperatures, polluted ground or rising ground water. After the shell is finished, the interior work has to be done. Within this phase, plaster may not dry, for example, which leads to delays or may be dried using construction dehydrators. Based on this given information, we planned both a regular, non-context-aware as well as a context-aware process model by means of our approach.

Thereafter, we presented both process models to the architect (in a next meeting) and questioned him on how he assesses the validity (Adelman, 1991) of the context-aware process model with respect to the context signals he told us before. The professional architect as well as several further experts in the area of process modelling supported the efficacy of our approach. They considered the resulting context-aware process model as valid as it is more suitable regarding the consideration of possibly occurring context signals within process models compared to the regular process model. By means of both, the prototypical implementation and the experimental evaluation we aimed to show that we are able to take process exogenous changes of context variables (due to environmental events) into account in process models of real world processes.

We examined the size of the non-context-aware process models (i.e., the number of actions and belief states) in contrast to the size of the context-aware process models (i.e., the number of actions, belief states and receive context actions; note that multiple planned receive context actions may be subsumed by means of Interruptible Activity Regions in Figure 2.7) as an indicator of the complexity of the planning problem on the one hand and for the resulting flexibility concerning environmental influences in general and context events in particular on the other hand. Large process models (like the House building process) may not be very suitable for visualizing processes but, however, they are particularly suitable as a foundation for process-aware information systems and especially for automated execution of processes by means of, for example, workflow-engines (cf., e.g., Khan et al., 2010; Weber, 2007). A summary of all measured properties and the regarding durations for planning both the regular and the context-aware process model can be seen in Table 2.2.

2.2.6 Conclusion, Limitations, and Further Research

Within this paper, we presented an approach construct complex and comprehensive context-aware process models in an automated way and therefore contribute to a heretofore unsolved issue. Thereby, we consider both static and non-static context within the regular control-flow of process models. By means of such context-aware process models we are able to shift complexity from runtime to design time, which may be useful

Process name	Number of distinct belief state variables, actions and context signals	Number of actions and belief states in the non-context-aware process model	Number of actions, belief states and receive context actions in the context-aware process model	Average duration for planning the non-context-aware process model	Average duration for planning the context-aware process model
Running example	6 / 3 / 2	5 / 4	26 / 22 / 11	0.16 s	0.19 s
Day city trip	9 / 12 / 10	86 / 46	3714 / 1228 / 2983	0.34 s	3.34 s
House building	29 / 30 / 8	59 / 37	8431 / 3659 / 3430	0.19 s	20.42 s

Table 2.2: Evaluation Results by means of a Prototypical Implementation

especially for performers without broad knowledge and expertise, and further enable the automated execution of context-aware processes by means of workflow-engines, for instance (cf., e.g., Khan et al., 2010). As we use well-known process modeling elements like Accept Event Actions and Interruptible Activity Regions within our approach, we ensure readability and understandability of the constructed context-aware process models. We further ensure the feasibility of our approach by means of mathematical proofs of its key properties, a prototypical implementation and the application to real world processes.

However, there are some limitations of our work, which have to be addressed in further research. Especially considering large processes like, for example, the House building process (cf. Section 2.2.5), our graphical notation by means of receive context actions implies very large resulting process models. As in process modeling notations like UML Activity diagrams and BMPN, belief states are not (explicitly) considered, further research is needed to increase readability of context-aware process models by means of, for instance, reducing the amount of required nodes within the process model. Some promising preliminary ideas tend to identify areas of (subsequent) actions in process models that could all be interrupted by one context signal. Thus, Interruptible Activity Regions as used in UML (*OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015) seem promising to reduce the number of receive context actions. We already strived this to some extent (as seen in Figure 2.7), but, as we wanted to focus on the construction of context-aware process models, we did not use a specific process modeling language. However, we already consider the issue of transferring the graph resulting from the nondeterministic transition system to a process modeling language within our research by means of automatically constructing control flow patterns and using UML activity diagrams (cf., e.g., Heinrich et al., 2009; Heinrich et al., 2012). Moreover, to enable the automated planning of context-aware process models in a user-friendly way, an existing process planning tool needs to be extended by the context-aware features.

Further, following the approach of Russell et al. (2006b), recovery strategies to gracefully handle the interruption of actions due to context signals may be considered. For example, an already started action (e.g., a cycling tour) may be rolled back (return home) or even continued if a suitable action is available and applicable (take on rain jacket). Basically, we are able to plan such recovery strategies by means of modeling the according recovery actions. However, it seems promising to support a modeler in terms of proposing potential recovery actions in a (semi-)automated way where possible.

Additionally, besides context events, other users could affect a user during the conduction of his/her process. As the consideration of multi-user processes within the research strand of automated planning is an unsolved issue, this would be a relevant next

step in order to fully cover adaptive process models regarding external influences regarding other users and context events.

2.2.7 Acknowledgements

The research was funded by the Austrian Science Fund (FWF): P 23546-G11.

2.2.8 References

Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). "Towards a Better Understanding of Context and Context-Awareness." In: *Handheld and Ubiquitous Computing*. Ed. by H.-W. Gellersen. Vol. 1707. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 304–307. ISBN: 978-3-540-66550-2. DOI: 10.1007/3-540-48157-5_29.

Adelman, L. (1991). "Experiments, Quasi-Experiments, and Case Studies: A Review of Empirical Methods for Evaluating Decision Support Systems." In: *IEEE Transactions on Systems, Man, and Cybernetics* 21.2, pp. 293–301.

Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.

Attard, J., Scerri, S., Rivera, I., and Handschuh, S. (Sept. 2013). "Ontology-based situation recognition for context-aware systems." In: *Proceedings of the 9th International Conference on Semantic Systems*. ACM, pp. 113–120. ISBN: 1450319726. DOI: 10.1145/2506182.2506197.

Ayora, C., Torres, V., Reichert, M., Weber, B., and Pelechano, V. (2013). "Towards Runtime Flexibility for Process Families: Open Issues and Research Challenges." In: *Business Process Management Workshops Lecture Notes in Business Information Processing* 132, pp. 477–488.

Baldauf, M., Dustdar, S., and Rosenberg, F. (2007). "A survey on context-aware systems." In: *International Journal of Ad Hoc and Ubiquitous Computing* 2.4, pp. 263–277. ISSN: 1743-8225.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., and Riboni, D. (2010). "A survey of context modelling and reasoning techniques." In: *Pervasive and mobile computing* 6.2, pp. 161–180. DOI: 10.1016/j.pmcj.2009.06.002. URL: http://www.sciencedirect.com/science/article/pii/S1574119209000510.

Bider, I. (2005). "Masking flexibility behind rigidity: Notes on how much flexibility people are willing to cope with." In: *Proceedings of the CAiSE* 5, pp. 7–18.

Bucchiarone, A., Mezzina, C., and Pistore, M. (2013). "Captlang: a language for context-aware and adaptable business processes." In: *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*.

Business Process Model and Notation (BPMN): Version 2.0.2 (2013). URL: http://www.omg.org/spec/BPMN/2.0.2/PDF (visited on 10/16/2014).

Chen, H., Finin, T., and Joshi, A. (2003). "An Ontology for Context-Aware Pervasive Computing Environments." In: *The Knowledge Engineering Review* 18.3, pp. 197–207.

Cobham, A. (1965). "The intrinsic computational difficulty of functions." In: *Logic, Methodology, and Philosophy of Science II*.

Dey, A. K. (2001). "Understanding and using context." In: *Personal and ubiquitous computing* 5.1, pp. 4–7. ISSN: 1617-4909.

Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). "A survey of autonomic communications." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 1.2, pp. 223–259. ISSN: 1556-4665.

Fujii, K. and Suda, T. (2009). "Semantics-based context-aware dynamic service composition." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 4.2, p. 12. ISSN: 1556-4665.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Gottschalk, F. and La Rosa, M. (2010). "Process Configuration." In: *Modern Business Process Automation*. Ed. by A. H. M. ter Hofstede, W. M. P. van der Aalst, M. Adams, and N. Russell. Berlin Heidelberg: Springer, pp. 459–487. ISBN: 978-3-642-03122-9. DOI: 10.1007/978-3-642-03121-2_18.

Gottschalk, F., van der Aalst, W. M. P., and Jansen-Vullers, M. H. (2007). "Configurable process models—a foundational approach." In: *Reference Modeling*. Springer, pp. 59–77. ISBN: 3790819654.

Hallerbach, A., Bauer, T., and Reichert, M. (2008a). "Context-based configuration of process variants." In: *Proceeding of the 3rd International Workshop on Technologies for Context-Aware Business Process Management (TCoB 2008)*.

Hallerbach, A., Bauer, T., and Reichert, M. (2010). "Capturing variability in business process models: the Provop approach." In: *Journal of Software Maintenance and Evolution: Research and Practice* 22.6–7, pp. 519–546. ISSN: 1532-0618.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems (ICIS)*. Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., Lewerenz, L., and Zimmermann, S. (Mar. 2015a). "Quality of Service-aware Service Selection: A Novel Approach Considering Potential Service Failures and Non-Deterministic Service Values." In: *INFORMS Service Science, A Journal of the Institute for Operations Research and the Management Sciences* 7.1, pp. 48–69. DOI: 10.1287/serv.2015.0093.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik (MKWI)*. Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). "Design science in information systems research." In: *MIS Q* 28.1, pp. 75–105. ISSN: 0276-7783.

Hoffmann, J., Weber, I., and Kraft, F. M. (2009). "Planning@ sap: An application in business process management." In: *Proceedings of the 2nd International Scheduling and Planning Applications woRKshop (SPARK'09)*.

Hoffmann, J., Weber, I., and Kraft, F. M. (July 2012). "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management." In: *Journal of Artificial Intelligence Research* 44.1, pp. 587–632. DOI: 10.1613/jair.3636.

Hu, B., Wang, Z., and Dong, Q. (Oct. 2012). "A Modeling and Reasoning Approach Using Description Logic for Context-Aware Pervasive Computing: Emerging Research in Artificial Intelligence and Computational Intelligence." In: *Emerging Research in Artificial Intelligence and Computational Intelligence, Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence*. Ed. by J. Lei, F. L. Wang, H. Deng, and D. Miao. Vol. 315. Communications in Computer and Information Science.

Springer Berlin Heidelberg, pp. 155–165. ISBN: 978-3-642-34240-0. DOI: 10.1007/978-3-642-34240-0_21.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security (IJCSIS)* 7.2, pp. 147–152.

La Rosa, M., Dumas, M., ter Hofstede, A. H. M., and Mendling, J. (2011a). "Configurable multi-perspective business process models." In: *Information Systems* 36.2, pp. 313–340. ISSN: 0306-4379.

Mattos, T. d. C., Santoro, F. M., Revoredo, K., and Nunes, V. T. (2014). "A formal representation for context-aware business processes." In: *Computers in Industry* 65.8, pp. 1193–1214. ISSN: 0166-3615. DOI: 10.1016/j.compind.2014.07.005. URL: http://www.sciencedirect.com/science/article/pii/S0166361514001407.

Meyer, H. and Weske, M. (2006). "Automated service composition using heuristic search." In: *Business Process Management*, pp. 81–96.

Mulholland, A., Thomas, C. S., Kurchina, P., and Woods, D. (2006). *Mashup corporations: The end of business as usual*. Evolved Technologist Press.

Nadoveza, D. and Kiritsis, D. (2014). "Ontology-based approach for context modeling in enterprise applications." In: *Special Issue on The Role of Ontologies in Future Web-based Industrial Enterprises* 65.9, pp. 1218–1231. ISSN: 0166-3615. DOI: 10.1016/j.compind.2014.07.007. URL: http://www.sciencedirect.com/science/article/pii/S0166361514001420.

OMG Unified Modeling Language TM (OMG UML): Version 2.5 (2015). URL: http://www.omg.org/spec/UML/2.5 (visited on 05/04/2016).

Öztürk, P. and Aamodt, A. (1997). "Towards a Model of Context for Case-Based Diagnostic Problem Solving." In: *IN CONTEXT-97; PROCEEDINGS OF THE*, pp. 198–208.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2008). "A Design Science Research Methodology for Information Systems Research." In: *Journal of Management Information Systems* 24.3, pp. 45–78.

Regev, G., Bider, I., and Wegmann, A. (2007). "Defining business process flexibility with the help of invariants." In: *Software Process: Improvement and Practice* 12.1, pp. 65–79.

Reichert, M. U. and Weber, B. (2012). *Enabling flexibility in process-aware information systems: challenges, methods, technologies*. Springer. ISBN: 3642304095.

Rosemann, M., Recker, J. C., and Flender, C. (2008). "Contextualisation of business processes." In: *International Journal of Business Process Integration and Management* 3.1, pp. 47–60. ISSN: 1741-8771.

Rosemann, M., Recker, J. C., and Flender, C. (2010). "Designing context-aware business processes." In: *Systems Analysis and Design: People, Processes, and Projects. Armonk, NY: ME Sharpe, Inc*, pp. 53–74.

Rosemann, M. and van der Aalst, W. M. P. (2007). "A configurable reference modelling language." In: *Information Systems* 32.1, pp. 1–23. ISSN: 0306-4379.

Russell, N., van der Aalst, W. M. P., and ter Hofstede, A. H. M. (2006b). "Exception Handling Patterns." In: *Process-Aware Information Systems. Technical report, BPM Center Report BPM-06-04, BPMcenter. org*.

Sakurai, Y., Takada, K., Anisetti, M., Bellandi, V., Ceravolo, P., Damiani, E., and Tsuruta, S. (2012). "Toward sensor-based context aware systems." In: *Sensors* 12.1, pp. 632–649.

Satyanarayanan, M. (2001). "Pervasive computing: Vision and challenges." In: *Personal Communications, IEEE* 8.4, pp. 10–17.

Schonenberg, H., Mans, R., Russell, N., Mulyar, N., and van der Aalst, W. M. P. (2008). "Process flexibility: A survey of contemporary approaches." In: *Advances in Enterprise Engineering I*. Springer, pp. 16–30.

Simons, C. and Wirtz, G. (2007). "Modeling context in mobile distributed systems with the UML." In: *Visual Interactions in Software Artifacts* 18.4, pp. 420–439. ISSN: 1045-926X. DOI: 10.1016/j.jvlc.2007.07.001. URL: http://www.sciencedirect.com/science/article/pii/S1045926X07000377.

Soffer, P. (2005). "On the notion of flexibility in business processes." In: *Proceedings of the CAiSE* 5, pp. 35–42.

Strang, T. and Linnhoff-Popien, C. (2004). "A Context Modeling Survey." In: *Workshop Proceedings. First International Workshop on Advanced Context Modelling, Reasoning And Management at UbiComp*.

Swenson, K. D. (2010). "Mastering the Unpredictable." In: *How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done, Glossary*, pp. 314–317.

Sycara, K., Paolucci, M., Ankolekar, A., and Srinivasan, N. (2003). "Automated discovery, interaction and composition of semantic web services." In: *Web Semantics: Science, Services and Agents on the World Wide Web* 1.1, pp. 27–46. ISSN: 1570-8268.

Tealeb, A., Awad, A., and Galal-Edeen, G. (2014). "Context-Based Variant Generation of Business Process Models." In: *Enterprise, Business-Process and Information Systems Modeling*. Springer, pp. 363–377.

Van der Aalst, W. M. P., Dreiling, A., Gottschalk, F., Rosemann, M., and Jansen-Vullers, M. H. (2006). "Configurable process models as a basis for reference modeling." In: *Business Process Management Workshops*. Ed. by C. J. Bussler and A. Haller. Springer, pp. 512–518. ISBN: 3540325956.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Van der Aalst, W. M. P., Weijters, A. J. M. M., and Maruster, L. (2004). "Workflow mining: Discovering process models from event logs." In: *IEEE Transactions on Knowledge and Data Engineering* 16.9, pp. 1128–1142.

Venable, J. R., Pries-Heje, J., and Baskerville, R. (2012). "A Comprehensive Framework for Evaluation in Design Science Research." In: *Design Science Research in Information Systems. Advances in Theory and Practice*. Ed. by K. Peffers, M. A. Rothenberger, and B. Kuechler. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 423–438. ISBN: 978-3-642-29862-2.

Wang, J., Zeng, C., He, C., Hong, L., Zhou, L., Wong, R. K., and Tian, J. (2012). "Context-aware role mining for mobile service recommendation." In: *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. Trento and Italy: ACM, pp. 173–178. ISBN: 978-1-4503-0857-1. DOI: 10.1145/2245276.2245310.

Wang, X. H., Zhang, D. Q., Gu, T., and Pung, H. K. (2004). "Ontology based context modeling and reasoning using OWL." In: *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pp. 18–22.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Wetzstein, B., Ma, Z., Filipowska, A., Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., and Cicurel, L. (2007). "Semantic Business Process Management: A Lifecycle Based Requirements Analysis." In: *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007) in conjunction with the 3rd European Semantic Web Conference (ESWC 2007)*.

Ye, J., Dobson, S., and McKeever, S. (2012). "Situation identification techniques in pervasive computing: A review." In: *Pervasive and mobile computing* 8.1, pp. 36–66.

Zhang, D., Adipat, B., and Mowafi, Y. (2009). "User-Centered Context-Aware Mobile ApplicationsâThe Next Generation of Personal Mobile Computing." In: *Communications of the Association for Information Systems* 24.1, p. 3. ISSN: 1529-3181.

Zhou, J., Gilman, E., Palola, J., Riekki, J., Ylianttila, M., and Sun, J. (2011). "Context-aware pervasive service composition and its implementation." In: *Personal and ubiquitous computing* 15.3, pp. 291–303. ISSN: 1617-4909. DOI: 10.1007/s00779-010-0333-5.

Zhu, X., Recker, J., Zhu, G., Santoro, F. M., and Al-Mashari, M. (2014a). "Exploring location-dependency in process modeling." In: *Business Process Management Journal* 20.6.

Zhu, X., Zhu, G., Vanthienen, J., Baesens, B., et al. (2014b). "Towards location-aware process modeling and execution." In: *Business Process Management Workshops*.

2.3 Paper 3: The Cooperation of Multiple Actors within Process Models: An Automated Planning Approach

Status	Published	Full Citation
Accepted	04/2019	Heinrich, B., Schiller, A., Schön, D. (2019), The cooperation of multiple actors within process models: an automated planning approach, Journal of Decision Systems, 27:4, pp. 238-274. DOI: 10.1080/12460125.2019.1600894.

Abstract

In many business processes, multiple actors such as different employees, departments or companies are involved. These actors need to work together and form appropriate partnerships in parts of these processes to achieve their individual goals. Hence, from a Business Process Management perspective, the actors need to cooperate. We present a conceptual foundation for multi-actor process models, which enables the consideration of individual starting points and goals as well as partnerships. Further, we incorporate the cooperation of actors in the control flow of process models by constructing explicit actions determining where in the process to form and disband partnerships. We pursue an automated planning approach due to the complexity of the required cooperation. The constructed multi-actor process models are proven to be correct and complete. We demonstrate the feasibility of our approach by an application in several real-world scenarios and its effectiveness through the assessment of a practitioner.

Keywords: Business process management; Multi-actor processes; Process modeling

Post publication changes:

- Several formatting changes for consistency reasons
- Changed citation style for consistency reasons
- Changed spelling from British English to American English for consistency reasons
- Corrected line numberings in Listings and according references due to line breaks
- The numbering of headlines, tables, figures and definitions was changed for consistency reasons
- Consistent capitalization of Section references

2.3.1 Introduction

Ever-increasing competition in today's business world requires companies to reduce costs and increase their efficiency. Hence, companies need to consider economic effects of, for instance, strengthening their own capabilities in a particular business area (Forstner et al., 2014) to stay competitive. With companies focussing on particular capabilities, oftentimes multiple actors such as different departments, companies or individuals are involved in the conduction of a business process (cf.,e.g., Davenport et al., 1990; The Workflow Management Coalition Specification, 1999). These conducting actors *cooperate* by forming so-called *partnerships* – which means, sets of selected actors (cf. Grefen et al., 2000; Huang et al., 2013; Leymann et al., 2002) jointly conduct parts of the process (Pulgar et al., 2017). In a similar vein, Serve et al. (2002) state that 'business processes are linked and managed across multiple companies' as it could be beneficial for a company to source out parts of its business processes (Katzmarzik et al., 2012). In such inter-organizational processes, each conducting actor (e.g., suppliers, partnering companies or customers) usually starts at an individual starting point, follows its own individual goals (cf.,e.g., Becker et al., 2013; Chiu et al., 2003; Skjoett–Larsen et al., 2003) but needs to cooperate with other conducting actors (cf. Lambert et al., 1996; Stadtler et al., 2015). Similarly, multiple actors (e.g., departments or employees of a company) cooperate within intra-organizational processes (cf., e.g., Ghrab et al., 2017). In either case, the partnerships formed to jointly conduct actions during the process are usually not required throughout the entire process, but just for certain parts of it (Grefen et al., 2000). To give an example, customers can possibly conduct parts of a process on their own by using self-service technologies (Klier et al., 2016). A process comprising partnerships of actors that conduct parts of it jointly and in which actors start at an individual starting point and follow individual goals is referred to as a *multi-actor process* in the following.

Besides this discussion based on scientific literature, the relevance of multi-actor processes can also be reflected from a practical perspective. For instance, in cooperations with two European financial services providers (a bank and an insurance company) we supported an analysis of over 600 (core) processes from – amongst others – the divisions credit lending and securities trading (in case of the bank) and the general project management department (in case of the insurer). The aim of the cooperations was to increase transparency (e.g., definition of responsibilities) and efficiency regarding economic indicators and capacities of these processes. Therefore, detailed data for the processes themselves as well as the involved departments and actors was analysed. In this context, we examined – amongst other characteristics – which of these processes are multi-actor processes, which means, whether several actors in terms of employees and

departments of the financial services providers as well as external service providers and customers have to work together and thus form partnerships in parts of the processes. Our analyses showed the following results: Partnerships of at least two actors conduct parts of the process (i.e., actions) jointly in more than 90% of all considered insurer processes resp. more than 70% in case of the bank processes. Partnerships of three or more actors are comprised in more than 60% of the insurer processes resp. in more than 50% of the bank processes. Thereby, these actors do not necessarily represent individual employees but also departments that usually comprise more than one individual. Hence, the aforementioned sizes of the partnerships serve as lower bounds and, in a particular process execution, usually more individuals are involved. The examined partnerships were used for several reasons by both companies: For example, in many processes, they allowed a high utilization of resources and an efficient workload of employees. Furthermore, in some cases, they were required to ensure legal and regulatory compliance (e.g., to realize a dual or triple control principle). The security order management process of the bank may serve as an example for a multi-actor process: A number of brokers and order processing specialists, the internal risk assessor as well as external contractors are just some of the indispensable actors in this process to conduct actions jointly. This illustrates the motivation and importance of partnerships and jointly conducted actions in practice. Besides, we refer to the Section 2.3.5 Evaluation, where an evaluation of the approach provided in this paper by means of several of these processes is discussed and concrete key properties for the processes (e.g., the number of involved partnerships) are presented.

After discussing the relevance and importance of multi-actor processes in research and practice, we will focus on how multiple actors in process models are currently addressed within the research field of BPM (Business Process Management; e.g., Chinosi et al., 2012) as process models are an established way to represent processes. Within different well-known process modeling languages, concepts for representing multiple actors exist. For example, the languages BPMN and UML comprise so-called swimlanes that allow to associate actions to one specific actor that needs to conduct these actions (*Business Process Model and Notation* (*BPMN*)*: Version 2.0.2* 2013; *OMG Unified Modeling Language TM* (*OMG UML*)*: Version 2.5* 2015). These swimlanes merely serve as an annotation. However, the association of actions to conducting actors by swimlanes is not sufficient for reliably forming appropriate partnerships (cf., e.g., Kossak et al., 2016; Natschläger et al., 2013; Pulgar et al., 2017; Recker et al., 2006; Wohed et al., 2006). In particular, the lack of expressiveness is considered a major weakness of swimlanes (Kossak et al., 2016; Natschläger et al., 2013). In this context, lack of expressiveness means that it is hardly possible to express that an action needs to be performed by a particular number of selected actors jointly, which means to express the required size and composition

of a partnership. Pulgar et al. (2017) highlight this issue and state that 'there is no natural way to represent collaborative activities performed by different roles' (i.e., actors in our terms). Hence, in order to represent multi-actor processes with possibly individual starting points and goals for each actor (cf. Aspect (A1)) comprising actions performed by partnerships (each with a required size and composition; cf. Aspect (A2)) by means of established process modeling languages, a new approach needs to be developed. Further, from a decision support perspective, it is promising to support individual actors explicitly when and with whom to cooperate (Peleteiro et al., 2014). We therefore aim at incorporating the cooperation of multiple actors in the control flow perspective of process models (van der Aalst et al., 2003; van der Aalst et al., 2002) by planning explicit actions determining when to form and when to disband appropriate partnerships (Aspect (A3)). Constructing a multi-actor process model based on this conceptual foundation – instead of using annotations such as swimlanes mentioned above – is envisioned in order to increase the expressiveness of multi-actor process models. We therefore want to take the Aspects (A1) to (A3) into account and state our first research question of *how a conceptual foundation to represent multi-actor process models can be specified.*

Besides addressing Aspects (A1) to (A3), as process (re)design projects and process models are becoming increasingly large and complex (Hornung et al., 2007), constructing process models manually develops into a more and more difficult and error-prone task. More precisely, according to Mendling et al. (2008), larger (they refer to 40 actions and more) and more complex process models particularly tend to contain more errors when constructed manually. Empirical studies of Roy et al. (2014) and Fahland et al. (2011), for instance, show that up to 92.9 % of process models are erroneous in industrial contexts. Besides semantic errors (e.g., missing actions), in particular, syntactical errors such as hanging nodes and ambiguous gateways are contained in these process models. Even though these errors do not render the process models completely worthless, they make it very difficult to use the models for potential process improvements or to apply several approaches for the automated verification (Weber et al., 2008b) and execution (Khan et al., 2010; Weber, 2007), for instance. Further, compared to constructing single-actor process models, constructing multi-actor process models manually is even more complex and error-prone as it poses additional challenges (Aspects (A1) to (A3)). For example, individual starting points and goals, actors cooperating in partnerships and the size and composition of these partnerships need to be taken into account. We thus strive to address the construction of multi-actor process models *in an automated manner* and state our second research question of *how feasible multi-actor process models can be constructed by means of an automated planning approach*.

The second research question is in accordance with the emergence of several approaches to support modelers and business analysts by means of automation (e.g., algo-

rithms) in the last years. For instance, process mining (e.g., IEEE Task Force on Process Mining, 2012; van der Aalst, 2015; van der Aalst et al., 2004) assists business analysts especially in the *process analysis* phase of the BPM Lifecycle (cf. Wetzstein et al., 2007). Automated service selection and composition (e.g., Ding et al., 2015; Paik et al., 2014; Wang et al., 2014) increase the degree of automation within the phases *process implementation* and *process execution*. Our second research question falls within the research strand automated process planning, which envisions the construction of process models in an automated manner by means of algorithms (Heinrich et al., 2015b; Heinrich et al., 2015c; Henneberger et al., 2008; Hoffmann et al., 2012) to support modelers in the phase of *process modeling*.

To sum up, we present an approach for the *automated planning of multi-actor process models* (second research question) based on a *conceptual foundation* (first research question) that incorporates the cooperation of conducting actors. The main contributions are as follows:

❶ *Conceptual foundation for multi-actor process models* (*Aspects* (*A1*) *to* (*A3*)). We propose a conceptual foundation that enables the consideration of individual starting points and goals of conducting actors as well as partnerships that need to conduct actions jointly. These partnerships are of a particular size and consist of specific actors. The conceptual foundation further includes the cooperation of conducting actors within the control flow of process models. Cooperation is expressed by explicit actions denoting where and with whom to form and disband partnerships.

❷ *Automated planning of multi-actor process models.* We propose an automated planning approach, the first to support a construction of feasible, correct and complete multi-actor process models.

In the remainder of this paper we follow the research approach as presented by Bertrand et al. (2002) as well as Mitroff et al. (1974) and its phases conceptualization, modeling, model solving, and implementation: After this introduction of the problem context (conceptualization), we discuss related work in the next Section. Thereafter we introduce our planning domain and the running example we use to illustrate our approach. Subsequently, we present a conceptual foundation for multi-actor process models (i.e., a 'model of the object reality'; cf. Meredith et al., 1989; modeling) and discuss how the construction of multi-actor process models can be supported by means of the proposed automated planning approach (model solving by means of an algorithm). In the penultimate Section, we evaluate our approach in terms of its termination as well as the completeness and correctness of the constructed process models (i.e., 'proof of the solution'; cf. Bertrand et al., 2002). We further demonstrate the feasibility of our approach

by means of a software prototype (implementation) as well as an application to different real-world scenarios as proposed by Meredith et al. (1989). Moreover, we evaluate its effectiveness by an application in an experimental real-world scenario together with a practitioner. Thereby we aim at evaluating in how far our approach is able to construct multi-actor process models that reflect processes as actually conducted in reality according to the assessment of a practitioner. Finally, we summarize our considerations, discuss limitations and provide an outlook on future steps.

2.3.2 Related Work

In this Section, we give an overview of how different fields of research address multi-actor processes and the construction of multi-actor process models. We (1) introduce approaches dealing with multi-actor processes and workflows within the general field of BPM before (2) distinguishing existing approaches within the focussed research field of automated planning from ours. Finally, we briefly analyze the related areas process mining (3) and multi-agent-systems / autonomous systems (4).

Ad (1): Multi-actor processes are heavily discussed within the research field of BPM (cf., e.g., Chen et al., 2001; Fleischmann et al., 2013; Jennings et al., 2000; Kannengiesser, 2017).

To begin with, the swimlanes in modeling languages such as BPMN and UML allow the annotation of multi-actor processes (*Business Process Model and Notation* (*BPMN*)*: Version 2.0.2* 2013; *OMG Unified Modeling Language TM* (*OMG UML*)*: Version 2.5* 2015), but do not specify a conceptual foundation for multi-actor process models (cf. contribution ❶). Shapiro et al. (2012) discuss different possibilities to represent actions that need to be performed by partnerships by means of swimlanes. However, each of these possibilities has major shortcomings. For instance, one proposition is to duplicate actions in the swimlane for each actor that jointly conducts the action. This results in 'messy and difficult to understand' process models (Pulgar et al., 2017). In contrast, a cooperation of actors is also discussed in the research field of workflow management, where cross-organizational and collaborative workflows (cf. Boukhedouma et al., 2017; Liu et al., 2015; Liu et al., 2016; van der Aalst, 1999) are examined. Here, 'some tasks can only be executed by certain business partners and a case always resides at exactly one location' (van der Aalst, 1999). However, these works do not present a conceptual foundation for multi-actor process models (cf. contribution ❶). Further, they do not aim to plan multi-actor process models in an automated manner but mostly rely on already existing process models (cf., e.g., Boukhedouma et al., 2017), and thus do not address contribution ❷.

Other works in the field of BPM consider so-called agents (corresponding to 'actors'

in our sense) as (decentrally) acting entities, that need to interact with each other during the execution of a process in order to reach their individual goals (Jennings et al., 2000; Kannengiesser, 2017). In particular, these agents apply (bilateral) communication and collaboration during process execution to align their individual tasks with others. However, such a consideration during the execution of a process is not beneficial in all cases. For instance, processes that are conducted within a department of an organization or even inter-organizational processes usually are controlled, modeled and managed across the involved actors in order to 'reduce operating cost, improve customer service and expand into markets' (Serve et al., 2002). We therefore aim at constructing multi-actor process models (at design time; cf. contribution ❷) instead of a communication-based approach that takes place during the execution of a process.

Moreover, works in the research field of resource management deal with the task of automated team selection and allocation (cf., e.g., Cabanillas et al., 2015; Havur et al., 2015; Havur et al., 2016). For example, Cabanillas et al. discuss a language for the description of teams (corresponding to 'partnerships' in our sense) in process models, which partly focusses on Aspect (A2), but does not cope with Aspects (A1) and (A3) (cf. ❶). Particularly, they extend the 'organizational metamodel' (Russell et al., 2005) by means of team-related concepts such as 'TeamRoles' (i.e., the role a person has in a team) or sizes of teams. However, these approaches do not aim to construct process models (cf. ❷) and instead rely on a given process model.

Ad (2): Within the field of automated planning, several approaches dealing with the problem of planning in so-called multi-agent environments exist. A survey conducted by de Weerdt et al. (2009) classifies these approaches into two basic categories: Planning by multiple instances (i.e., different planners) and planning for multiple actors. Approaches considered in the first category strive to distribute the problem of planning among several instances (cf., e.g., Torreño et al., 2012), which is not in the scope of this paper. The approaches of the second category address the problem of planning for multiple actors in different ways. Dimopoulos et al. (2006) aim to coordinate two actors with individual plans (i.e., processes), where one actor has to provide his/her plan proposals to a second actor based on which the second actor has to construct non-conflicting (to the provided plan proposal) plans to achieve the goals of both actors. Nissim et al. (2010) address the problem of planning in multi-actor environments by means of single-actor planning approaches, conducted by each actor in a distributed manner. In a second step, the corresponding single-actor plans are matched by means of 'seeking sequences of public actions [i.e., single-actor plans] that satisfy a certain CSP [constraint satisfaction problem]' (Nissim et al., 2010). Other approaches (cf., e.g., Crosby et al., 2013; Ephrati et al., 1994) deal with the problem by decomposing a general multi-actor process into smaller, single-actor processes in a first step and conducting a single-actor

planning for each of those subparts in a second step. This is also envisioned within the research field of (web) service composition (cf. Falou et al., 2009). However, none of these approaches aims to provide a conceptual foundation for multi-actor process models (cf. contribution ❶), which makes them substantially different from ours. Further planning approaches considering multiple actors exist, but the planning domain they use is fundamentally different from the planning domain needed for the automated planning of process models, for instance, by not considering actor-specific goal states (Torreño et al., 2014a). To give another example, Chouhan et al. (2017) address the issue of different actors having to perform different actions simultaneously to achieve a common goal and hence conduct a planning approach to "synchronize" these actors. In contrast, on the one hand, we aim at the cooperation of different actors performing one or more actions (or parts of processes) jointly instead of actors performing their actions separately but simultaneously. On the other hand, the planning domain they use does not aim to cope with actor-specific goal states as well, which is needed for the automated planning of multi-actor process models. Moreover, some of these approaches are – additionally to their different aims – based on heuristic techniques and do not provide a complete solution, which means, the constructed graphs do not contain all feasible paths (Štolba et al., 2013; Torreño et al., 2014b). Hence, these approaches do not fit the needs of automated planning of process models.

Ad (3): The research field of process mining addresses, amongst others, the issue of reconstructing process models from event logs in an automated manner. Here, as well, approaches to reconstruct models of processes with multiple actors (cf., e.g., Rozinat et al., 2009; Ou-Yang et al., 2011) or process models with a consideration of 'resources' executing the tasks (Schönig et al., 2015) exist. However, process mining follows an *as-is* perspective (cf. Rosemann et al., 2015) by reconstructing process models that denote already implemented and executed processes. In contrast, automated planning follows a *to-be* perspective as it strives to construct new process models (cf. contribution ❷). Furthermore, existing approaches do not aim at providing a conceptual foundation for multi-actor process models (cf. contribution ❶).

Ad (4): The research fields of multi-agent-systems (Shoham et al., 1995; Wooldridge, 2009; Zhang, 2017) and autonomous systems (Dobson et al., 2006) aim to address the cooperation of multiple agents (in our terms, 'actors') in processes. They do so during the execution of a process based on communication between the actors or between actors and a central coordination mechanism, which is a related but different task (cf. contributions ❶ and ❷).

To sum up: There are several valuable contributions regarding the consideration of multiple actors in process models in the literature (cf. Table 2.3).

However, none of these works aim to incorporate the cooperation of multiple actors

Research field	Analysed related work	Phase of consideration	Conceptual foundation for multi-actor process models (cf. ❶)			Automated construction of multi-actor process models (cf. ❷)
			Consideration of individual starting points and goals (cf. Aspect (A1))	Consideration of required size and composition of partnerships (cf. Aspect (A2))	Incorporation of formation and disbandment of partnerships in control flow (cf. Aspect (A3))	
BPM (general)	Chen et al., 2001; Fleischmann et al., 2013; Jennings et al., 2000; Kannengiesser, 2017	Design time and execution time	Not considered	Not considered	Not considered	Not considered
Process modeling languages	*Business Process Model and Notation (BPMN): Version 2.0.2* 2013; *OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015; Shapiro et al., 2012	Design time	Not considered	Not considered	Not considered	Not considered
Workflow management	Boukhedouma et al., 2017; Liu et al., 2015; Liu et al., 2016; van der Aalst, 1999	Execution time	Not considered	Not considered	Not considered	Not considered; relying on already existing process models
Resource management	Cabanillas et al., 2015; Havur et al., 2015; Havur et al., 2016	Execution time	Not considered	Partly considered	Not considered	Not considered; relying on already existing process models
Automated planning & Web service composition	Chouhan et al., 2017; Crosby et al., 2013; Dimopoulos et al., 2006; Ephrati et al., 1994; Falou et al., 2009; Nissim et al., 2010; Torreño et al., 2014a	Design time	Not considered	Not considered	Not considered	Partly considered (e.g., heuristic approaches; different planning domains)
Process mining	Rozinat et al., 2009; Ou-Yang et al., 2011	Design time	Not considered	Not considered	Not considered	Considered but as-is perspective (i.e., reconstructing models of already executed processes)
Multi-agent-systems	Dobson et al., 2006; Shoham et al., 1995; Wooldridge, 2009; Zhang, 2017	Execution time	Considered	Not considered	Not considered	Not considered

Table 2.3: Overview of Related Work

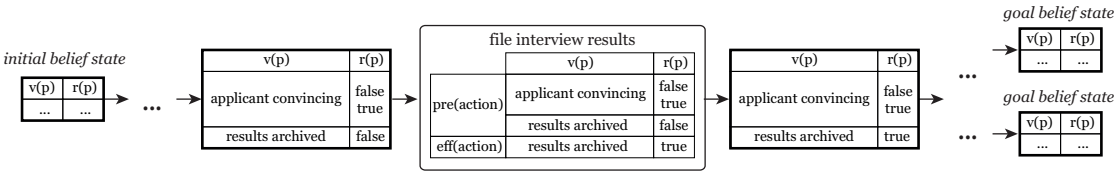

Figure 2.8: Illustrating the Action *file interview results* in the Running Example

based on a conceptual foundation (cf. contribution ❶) in process models in an automated manner (cf. contribution ❷).

2.3.3 Planning Domain

Within this Section, we introduce the planning domain, which we will extend to cope with multi-actor planning in the remainder of the paper. The automated construction of process models can be understood as a planning problem (e.g., Heinrich et al., 2009). More precisely, we have to abstract from an individual process execution and its world states in order to construct entire process models, valid for various process executions, resulting in a nondeterministic planning problem with belief states (Ghallab et al., 2004). Here, a belief state represents possibly infinite sets of world states. Hence, we use a general set-theoretic planning domain (cf. Ghallab et al., 2004; Ghallab et al., 2016) independent of a concrete representation language (e.g., process modeling language) for our approach. This ensures a maximum of compatibility with existing approaches in the literature (e.g., Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2015b; Heinrich et al., 2016; Meyer et al., 2006; Sycara et al., 2003) and enables a widespread use of our approach to construct multi-actor process models. Considering this planning domain, a bipartite *planning graph,* which consists of two types of nodes - representing *belief states* and *actions* - and edges is used.

To illustrate our approach and the planning domain, we will use an excerpt of a real-world human resources process at a university with several participating actors that need to cooperate. In this process, one of the two research project managers (*Bob* and *Danielle*), in a first step, checks the application documents sent by an applicant. If the application documents meet the requirements, the action *job interview* is conducted. Here, the personnel officer (*Eric*), the two research project managers and one additional (but not mandatory) chair member (*Silvia*) interview the applicant jointly. Further, if the applicant was convincing and the salary requirements of *Eric* and the applicant fit, s/he is engaged. In a next step, the results of the interview are filed. We will use this action *file interview results* (denoted by a rounded rectangle; belief states denoted by tables with a bold border) to illustrate the core concepts of the planning domain (cf. Figure 2.8).

A belief state *bs* can be seen as a set of information about the variables currently available in a process state (so-called *belief state variables*). A belief state is a set of *belief state tuples* (denoted by rows in the tables in Figure 2.8), each of which denotes one particular characteristic. For instance, the belief state tuple (*results archived,* {*false*}) in the belief state before the action *file interview results* expresses that at this state in the process, the results have not yet been archived.

Definition 2.3.1. (*belief state tuple*). A *belief state tuple p* is a tuple consisting of a *belief state variable* $v(p)$ and a subset $r(p)$ of its predefined *domain* $dom(p)$, which we will write as $p{:=}(v(p),r(p))$. The domain, $dom(p)$, specifies which values can generally be assigned to $v(p)$. The set $r(p) \subseteq dom(p)$ is called the *restriction* of $v(p)$ and contains the values that can be assigned to $v(p)$ in this specific belief state tuple p.

Definition 2.3.2. (*belief state*). A belief state *bs* is a finite set of belief state tuples, containing every belief state variable one time at the most. In the following, *BS* is a finite set of belief states.

To represent *actions* (denoted by rounded rectangles) conducted by an actor during a process, a second type of node is defined:

Definition 2.3.3. (*action*). Let BST be a finite set of belief state tuples. An action $action$ is a triple consisting of the action name and two sets, which we write as $action := (name(action), pre(action), eff(action))$. The set $pre(action) \subseteq BST$ are the preconditions of the action $action$, which describe the circumstances under which $action$ can be applied and the set $eff(action) \subseteq BST$ are the effects of the action $action$, denoting the consequences that result from applying *action*. In the following, $ACTIONS$ is a finite set of actions.

Definition 2.3.4. (*applicability*). An action $action$ is *applicable* in a belief state bs iff

$$\forall\, p \in pre(action)\, \exists\, q \in bs : v(p) = v(q) \;\wedge\; r(p) \cap r(q) \neq \varnothing$$

In other words, $action$ is applicable in bs iff all belief state variables in $pre(action)$ also exist in bs and the respective restrictions of the belief state variables intersect.

The preconditions and effects of actions are denoted by the table underneath the action name (cf. Figure 2.8). The action *file interview results* is applicable if the belief state variable *applicant convincing* is either *false* or *true* and the belief state variable *results archived* is *false* in the previous belief state. Its effects set the belief state variable *results archived* to *true*. Based on Definitions 2.3.1-2.3.4, a planning graph can be generated by means of different existing algorithms that progress from an initial belief state

to goal belief states (see, e.g., Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2009). To each action *action* applicable in a belief state *bs* a state transition function $R(bs, action)$ associates the next belief state. We define our planning graph as follows:

Definition 2.3.5. (*planning graph*). A planning graph is a bipartite, directed, finite graph $G=(NODES, EDGES)$, with the set of nodes *NODES* and the set of edges *EDGES*. The set of nodes *NODES* consists of two partitions: The set of action nodes *ACTIONS* and the set of belief state nodes *BS*. Each node $bs \in BS$ represents one distinct belief state in the planning graph. The planning graph starts with one initial belief state $Init \in BS$ and ends with one to possibly many goal belief states $Goal_j \in BS$.

As many real-world processes use large data types (e.g., many of the processes of the financial services providers mentioned in the introduction), a possibly infinite set of different process instances may exist. The above presented planning domain supports this subject, in contrast to STRIPS (Fikes et al., 1971) and other planners based on a classical planning framework (e.g., Hoffmann et al., 2012; for a detailed discussion of this aspect, cf. Heinrich et al., 2015b). However, as our approach extends existing single-actor planning approaches based on the introduced, common planning domain, existing works for the automated construction of control flow patterns within single-actor process models (e.g., Heinrich et al., 2015a; Heinrich et al., 2016; Meyer et al., 2006) can be further used for this purpose. Thus, it is not necessary to address how to incorporate control flow patterns (van der Aalst et al., 2003) such as exclusive choice into multi-actor process models in this paper. Following this, we do not consider control flow patterns in our running example as well.

2.3.4 Approach to construct multi-actor Process Models

We divide the overall goal of an automated planning of multi-actor process models (cf. contribution ❷) into sub goals in accordance with the previously discussed Aspects (A1) to (A3). At first, we extend the introduced planning domain to cope with actor-specific information. Thereafter, as actions may need to be conducted by partnerships (i.e., sets of selected actors, each with a particular size), we include cardinalities (i.e., the size of these partnerships) in the planning domain. Subsequently, we outline how to enable the cooperation of multiple actors (cf. contribution ❶). We will discuss each of these three sub goals:

(1) *Consider actor-specific information within the planning domain.* To consider actor-specific information in our planning domain, we adapt Definitions 2.3.1-2.3.5. In particular, we extend the definition of belief state tuples to denote actor-specific variables in terms of a set-theoretic representation.

(2) *Consider cardinalities.* Actions in a multi-actor process may need to be conducted by a certain number of actors. Therefore, we extend the definition of actions to represent a condition regarding the cardinality of a partnership, which is required to conduct a certain action.

(3) *Plan partnerships of actors.* As actions in a multi-actor process may be required to be conducted jointly by multiple actors represented within different belief states, we propose the join of multiple belief states into one belief state, representing a partnership. Similarly, actions may be required to be conducted by a subset of the actors represented within a single belief state. Thus, we describe how to split belief states into multiple belief states with those subsets of actors. In this way, the envisioned concept for enabling the cooperation of actors by *explicit* actions is addressed.

After explicating these sub goals (1) to (3) in more detail in the next subsections, we present our algorithm for the automated planning of multi-actor process models in a final subsection.

2.3.4.1 Consider actor-specific Information within the Planning Domain

As a first step, we describe how to represent actor-specific information in terms of the planning domain. Within the planning domain given by Definitions 2.3.1 to 2.3.5, there is no differentiation between *non-actor-specific* belief state variables and *actor-specific* belief state variables. Thus, there is no way to describe belief state variables related to a certain actor. This is insufficient for planning multi-actor processes: Not only information unrelated to a specific actor such as for example the availability of general resources or general process conditions is required in order to fully characterize the current process situation by means of belief states. Rather, actor-specific information such as the present status or capabilities of an actor needs to be included as well. Hence, we adapt the planning domain described in the previous Section and distinguish between actor-specific and non-actor-specific belief state variables. To be more precise, we extend the previous definition of belief state tuples by a so-called *actor specification* $a(p)$ with $a(p) \subseteq ACTORS \cup \{non\text{-}actor\} \cup \{arbitrary\}$. Here, *ACTORS* represents the set of actors participating in the conduction of the process, {*non-actor*} serves as an identifier for non-actor-specific variables and {*arbitrary*} denotes not mandatory actor-specific belief state tuples which will be discussed later in this subsection. Formally, the extended definition of belief state tuples (cf. Definition 2.3.1) is as follows:

Definition 2.3.6. (*belief state tuple*). Let $ACTORS$ be a finite set of actors participating in the conduction of a process. A belief state tuple p is a tuple of a *belief state variable* $v(p)$, its

restriction $r(p)$, a subset of its predefined domain $dom(p)$, and the *actor specification* $a(p)$, which is written as $(v(p), r(p), a(p))$. The actor specification $a(p)$ is $\{non - actor\}$ for non-actor-specific variables and $\{arbitrary\}$ or a subset of $ACTORS$ with $\varnothing \neq a(p) \subseteq ACTORS$ for actor-specific belief state variables.

Following this, we adapt the previous definition of belief states (cf. Definition 2.3.2) as well.

Definition 2.3.7. (*belief state*). A belief state *bs* is a finite set of belief state tuples such that for all $p = (v(p), r(p), a(p)) \in bs : (a(p) \subseteq ACTORS \wedge \nexists q \neq p \in bs : v(p) = v(q), r(p) = r(q) \wedge \nexists q \neq p \in bs : v(p) = v(q), a(p) \cap a(q) \neq \varnothing) \vee (a(p) = \{non - actor\} \wedge \nexists q \neq p \in bs : v(p) = v(q))$. *BS* is a finite set of belief states.

Definition 2.3.7 takes into account that the restrictions $(r(p))$ of the same belief state variable $(v(p))$ may differ for multiple actors $(a(p))$ in a belief state: While Definition 2.3.2 states that a belief state contains 'every belief state variable one time at the most', this limitation has been adjusted appropriately in Definition 2.3.7. For instance, in the context of our running example, if *Eric* has already conducted the job interview with the applicant whereas *Bob* and *Danielle* have not (actor-specific) and the contract is not closed yet (non-actor-specific), the according belief state to represent this situation is as follows: *bs* = {(*applicant interviewed*, {*true*}, {*Eric*}), (*applicant interviewed*, {*false*}, {*Bob*, *Danielle*}), (*contract closed*, {*false*}, {*non-actor*})}.

According to Definition 2.3.3, an action consists of its name, its preconditions – which comprise everything an action requires to be applied, including input parameters – and its effects, which denote how the application of an action affects the state of the world, including output parameters. Both preconditions and effects consist of belief state tuples. In light of Definition 2.3.6, these preconditions and effects can now contain not only non-actor-specific variables, but also incorporate actor-specific variables. To be more precise, by defining actor-specific preconditions (i.e., belief state tuples with *a*(*p*)⊆*ACTORS* within the preconditions) it is possible to limit the applicability of an action to certain actors or to describe actor-specific conditions. To give an example, the restriction of the belief state variable *applicant interviewed* must be *false* for *Eric* and the two research project managers (*Bob* and *Danielle*) in a belief state in order to be able to apply the action *job interview*. To give another example, in one of the processes of the aforementioned insurance company, a project completion report is prepared. This report has to be approved by the client and the internal project manager of the insurance company jointly. However, the project manager previously prepares the report. Hence, in order to apply the action *approve project completion report*, the belief state variable *project report* has to be *prepared* for the project manager and *not approved* for the client. Similarly,

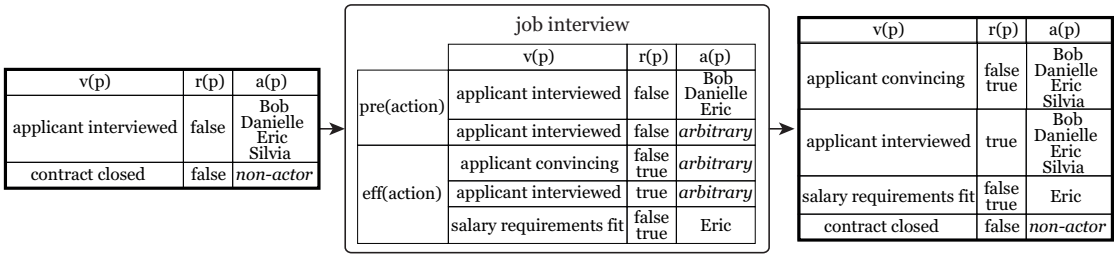

Figure 2.9: Illustrating the Action *job interview* in the Running Example

actor-specific effects (i.e., belief state tuples with $a(p) \subseteq ACTORS$ within the effects) allow to express actor-specific postconditions and, if needed, to limit the effects of an action to belief state variables referring to particular actors. Within our running example, this enables to denote that *Eric* and the applicant discuss the *salary requirements* during the job interview and *Eric* decides whether these requirements fit (cf. belief state tuple (*salary requirements fit*, {*false*, *true*}, {*Eric*})).

When preconditions and effects of actions are equivalent for different actors or partnerships (i.e., their belief state variables and restrictions are the same), it is preferable to not model explicit 'personalized' actions (which would differ only in the actor specification and possibly the name) for each actor or partnership. Instead, the amount of required specified actions can be reduced by means of generalization. This reduces the manual effort for the modeler and is more intuitive. For instance, within our example, the action *job interview* basically is the same task whether *Silvia*, the (not mandatory) chair member, participates or not. Hence, it should be modeled as one generic action (cf. Figure 2.9) instead of several actions which in each case explicitly include all selected actors (i.e., one action in which *Silvia* participates and one action in which she does not).

The actor specification of belief state tuples within the preconditions and effects of actions enables us to cope with this challenge: For this purpose, actor-specific belief state tuples with $a(p)$={*arbitrary*} can be used within the preconditions and effects of actions. This actor specification represents preconditions and effects that concern all actors conducting the action for which no other explicit precondition or effect (by means of a belief state tuple q with $v(p)$=$v(q)$, $a(q) \subseteq ACTORS$) is specified. To give an example, the aforementioned action *approve project completion report* comprises the effect {(*project report*, {*approved*, *not approved*}, {*arbitrary*})} as the belief state variable *project report* is either *approved* or *not approved* for both actors, the project manager as well as the client.

The definition of applicability (cf. Definition 2.3.4) is also adapted in order to take actor-specific variables in belief state tuples into account. For non-actor-specific variables ($a(p)$={*non-actor*}) in the preconditions of an action the applicability check re-

mains as specified in Definition 2.3.4. For belief state variables with $a(p) \subseteq ACTORS$ (e.g., (*applicant interviewed*, {*false*}, {*Bob, Danielle, Eric*})), it additionally needs to be checked whether all actors defined in the actor specification are represented in the belief state, the according actor-specific belief state variable exists and the restrictions (here: {*false*}) intersect. For belief state tuples with $a(p) = \{arbitrary\}$ (e.g., (*applicant interviewed*, {*false*}, {*arbitrary*})), the restriction (here: {*false*}) needs to be checked for all actors in the belief state that are not affected by an according actor-specific precondition (here: *Silvia*). Formally, the extended definition is as follows:

Definition 2.3.8. (*applicability*). Let $A\,(\text{bs}) := \bigcup_{(v(p),r(p),a(p)) \in bs | a(p) \subseteq ACTORS} a(p)$ be the set of all actors in a belief state bs. An action *action* is *applicable* in *bs* iff the following criteria are fulfilled:

- for all $(v(q), r(q), a(q)) \in pre(action)$ with $a(q) = \{non - actor\}$ there is a $(v(p), r(p), a(p)) \in bs$ such that $v(p) = v(q) and r(p) \cap r(q) \neq \varnothing$.
- for all $(v(q), r(q), a(q)) \in pre(action)$ with $a(q) \subseteq ACTORS$ it holds: For all actors $a \in a(q)$ there is a $(v(p), r(p), a(p)) \in bs$ such that $v(p) = v(q), a \in a(p)$ and $r(p) \cap r(q) \neq \varnothing$.
- for all $(v(q), r(q), a(q)) \in pre(action)$ with $a(q) = \{arbitrary\}$ it holds: For all actors $a \in A(bs) \setminus \{a' \in ACTORS \,|\exists (v(x), r(x), a(x) \in pre(action)$ such that $v(x) = v(q), a(x) \subseteq ACTORS, a' \in a(x)\}$ there is a $(v(p), r(p), a(p)) \in bs$ such that $v(p) = v(q), a \in a(p)$ and $r(p) \cap r(q) \neq \varnothing$.

To obtain a planning graph containing information about actors in the belief states, actor-specific effects (i.e., belief state tuples with $a(p) \subseteq ACTORS$ or $a(p) = \{arbitrary\}$ in the effects of actions) are applied when performing the state transition. Thus, belief state tuples p with $a(p) \subseteq ACTORS$ (e.g., (*salary requirements fit*, {*false, true*}, {*Eric*})) within the effects of an action are included in the belief state after the action. Further, for each belief state tuple p with $a(p) = \{arbitrary\}$ (e.g., (*applicant interviewed*, {*true*}, {*arbitrary*})), the belief state tuple $(v(p), r(p), A(bs) \setminus \{a' \in ACTORS \,|\exists (v(x), r(x), a(x)) \in eff(action)$ such that $v(x) = v(p), a(x) \subseteq ACTORS, a' \in a(x)\})$ is included in the belief state after the action. This guarantees that the respective effect is applied for each participating actor for which no contrary actor-specific belief state tuple is contained in the effects. By applying these actor-specific effects to the belief state on the left of Figure 2.9, the belief state tuple (*applicant interviewed*, {*true*}, {*Bob, Danielle, Eric, Silvia*}) is included in the belief state after the action *job interview* as can be seen on the right of Figure 2.9.

Each actor involved in a multi-actor process may start at an individual starting point and tends to reach individual goals (Aspect (A1); cf., e.g., Becker et al., 2013; Chiu et al., 2003; Skjoett–Larsen et al., 2003). Therefore, to construct feasible multi-actor process models, we take actor-specific initial states and goal states into account. For instance, in our running example, the goal of the research project managers and the chair member is to hire a new employee who has great professional expertise and integrates well into the team, whereas the personnel officer *Eric* requires the salary expectations of a new employee to fit into the budget. To give another example, in the project completion report process of the insurance company, the individual goal state of the project team is reached as soon as the final project meeting took place but the project manager has to conduct several further actions such as the preparation of the final report. To consider actor-specific initial states (which include actor-specific belief state tuples of only one actor) and goal states, the definition of a planning graph (cf. Definition 2.3.5) needs to be adapted:

Definition 2.3.9. (*planning graph*). A planning graph is a bipartite, directed, finite graph $G = (NODES, EDGES)$, with the set of nodes $NODES$ and the set of edges $EDGES$. The set of nodes $NODES$ consists of two partitions: The set of action nodes $ACTIONS$ and the set of belief state nodes BS. Each node $bs \in BS$ represents one distinct belief state in the planning graph. The planning graph starts with one to possibly many initial belief states $Init_i \in BS$ (one for each participating actor) and ends with one to possibly many goal belief states $Goal_j \in BS$, in which the goals of at least one actor are fulfilled.

2.3.4.2 Consider Cardinalities

We have just outlined how to consider actor-specific information in the planning domain. However, an important characteristic of multi-actor processes has not yet been addressed: Actions may potentially need to be conducted by a certain number of actors (Aspect (A2)). For instance, by means of the previous definition of the action *job interview*, it is only determined that the actor-specific variable *applicant interviewed* needs to have the restriction *false* for *Bob*, *Danielle*, *Eric* and all further actors conducting the action. However, it is not clear whether the action is supposed to be conducted by *Eric* and the two research project managers without an additional chair member or with a certain number of additional chair members. Thus, we extend the common definition of an action (cf. Definition 2.3.3) by including the cardinality of the partnership (i.e., set of actors) that has to conduct the action. The cardinality can be defined as a subset of the natural numbers. This definition reduces the amount of specification effort: It enables to specify actions that can be conducted by partnerships of different sizes (or even by a

single actor) in one single action, instead of having to specify each of these possibilities (i.e., for each feasible subset of actors) separately. We adapt Definition 2.3.3 as follows:

Definition 2.3.10. (*action*). Let BST be a finite set of belief state tuples. An action *action* is a quadruple $(name(action), cardinality(action), pre(action), eff(action))$ consisting of the action name $name(action)$, the set $cardinality(action) \subset \mathbb{N}$ denoting the possible sizes of partnerships required to conduct the action, the set $pre(action) \subseteq BST$ of preconditions of *action* and the set $eff(action) \subseteq BST$ of effects of *action*. It must hold $min(cardinality(action)) \geq \left|\bigcup_{(v(p),r(p),a(p)) \in pre(action) | a(p) \subseteq ACTORS} a(p)\right|$. In the following, *ACTIONS* is a finite set of actions.

In our example, the action *job interview* has to be conducted by at least *Eric* jointly with two research project managers *Bob* and *Danielle*, hence the cardinality of a partnership required to conduct the action *job interview* has to be at least 3. As a (not mandatory) additional chair member may or may not conduct the interview jointly with *Eric* and the two research project managers, the action should be applicable if the cardinality of the partnership is either 3 or 4. Thus, *cardinality*(*action*) = {*3,4*} is included in the definition of the action *job interview*.

The cardinality now needs to be considered in the applicability definition (cf. Definition 2.3.8) in order to ensure that actions are only applied in a belief state if their cardinality is met.

Definition 2.3.11. (*applicability*). Let $A(\text{bs}) := \bigcup_{(v(p),r(p),a(p)) \in bs | a(p) \subseteq ACTORS} a(p)$ be the set of all actors in a belief state bs. An action *action* is *applicable* in *bs* iff the following criteria are fulfilled:

- $|A(bs)| \in cardinality(action)$
- for all $(v(q), r(q), a(q)) \in pre(action)$ with $a(q) = non-actor$ there is a $(v(p), r(p), a(p)) \in bs$ such that $v(p) = v(q)$ and $r(p) \cap r(q) \neq \varnothing$.
- for all $(v(q), r(q), a(q)) \in pre(action)$ with $a(q) \subseteq ACTORS$ it holds: For all actors $a \in a(q)$ there is a $(v(p), r(p), a(p)) \in bs$ such that $v(p) = v(q)$, $a \in a(q)$ and $r(p) \cap r(q) \neq \varnothing$.
- for all $(v(q), r(q), a(q)) \in pre(action)$ with $a(q) = arbitrary$ it holds: For all actors $a \in bs \{a' \in ACTORS \,|\exists (v(x), r(x), a(x)) \in pre(action)$ such that $v(x) = v(q), a(x) \subseteq ACTORS, a' \in a(x)\}$ there is a $(v(p), r(p), a(p)) \in bs$ such that $v(p) = v(q), a \in a(p)$ and $r(p) \cap r(q) \neq \varnothing$.

By Definition 2.3.11 – compared to the common Definition 2.3.4 – the requirements for the planning of multi-actor process models are addressed by enabling to consider actor-specific belief state variables as well as the number of actors that has to conduct an action jointly.

2.3.4.3 Plan Partnerships of Actors

To complete the conceptual foundation for multi-actor process models and thus to fully address contribution ❶, we describe how to form and disband partnerships in the context of planning multi-actor process models and thus enable the cooperation of multiple actors by explicit actions.

As described in Definition 2.3.9 of the planning graph, for each actor $a_i \in ACTORS$ an individual initial state $Init_i \in BS$ with $A(Init_i)=\{a_i\}$ may be specified so that actions are planned from each of these individual starting points. However, in a multi-actor process, it is likely that an action (e.g., the action *job interview* from our running example) can or even needs to be applied jointly by multiple actors. Formally, this may happen due to actor-specific preconditions or cardinality restrictions. Hence, all actors conducting the process (e.g., *Bob*, *Danielle, Eric* and *Silvia*) need to be taken into account with regard to forming and disbanding partnerships, particularly in order to enable an application of actions that require specific actors and/or a specific number of actors. Partnerships can be seen as a set of actors represented by means of one, joint belief state. Thus, joining multiple, for example single-actor belief states (belief states with $|A(bs)|=1$), into one multi-actor belief state (a belief state with $|A(bs)| > 1$) is required. Within our running example, the individual initial states of *Bob*, *Danielle, Eric* and *Silvia* need to be joined in order to construct a joint belief state, so that the action *job interview* is applicable in this joint belief state. Additionally, in a multi-actor process, the situation can arise that only a subset of actors in an existing partnership can conduct an action jointly (e.g., due to an upper bound in the cardinality). We thus need to be able to disband a partnership and to split a belief state *bs* into a set of 'sub' belief states, each containing a subset of actors in *bs*. To enable the automated construction of a complete process model in which all appropriate partnerships are considered and hence to enable supporting individual actors when and with whom to cooperate, we address these issues by automatically identifying possibilities for forming and disbanding partnerships. We will construct respective *join actions* and *split actions* by means of an algorithm (cf. next subsection) and in compliance with the planning domain: These join actions and split actions are defined in terms of name, cardinality, preconditions and effects just like regular actions (cf. Definition 2.3.10), and the joint/split states result from the application of the join/split actions and their effects on the preceding states. Thus, by planning these

explicit actions we incorporate the formation and disbandment of appropriate partnerships in the control flow of the constructed process models (cf. Aspect (A3)) while ensuring compatibility with existing single-actor planning approaches.

The preconditions of a join/split action *action* are determined based on the according preceding belief state *bs* so that *action* is applicable in *bs*. Their cardinality is set to $|A(bs)|$. The effects of a join action are constructed so that all according belief state tuples for actors in the other (to be joined) belief states are added (i.e., created) by means of the effects. For instance, the effects of the join action expressing that *Eric* cooperates with *Bob, Danielle* and *Silvia* are defined as {(*applicant interviewed*, {*false*}, {*Bob, Danielle, Silvia*})} so that the joint state bs_{joined}= {(*applicant interviewed*, {*false*}, {*Eric, Bob, Danielle, Silvia*})} results from its application in the belief state bs_{Eric}= {(*applicant interviewed*, {*false*}, {*Eric*})}. The effects of a split action are specified contrarily, removing actor-specific belief state tuples of actors that are no longer part of the partnership after the disbandment.

Further, we aim at constructing join/split actions only when appropriate (i.e., feasible and necessary). Thus, we need to determine which belief states are appropriate for being *joined* and which belief states need to be *split*.

In a first step, we need to ensure that forming a partnership (i.e., joining a set of preceding belief states $\{bs_1, \ldots, bs_n\}$) is feasible and does not lead to logical contradictions. When forming a partnership, the status of each actor participating in the partnership must not be represented by more than one of the belief states. Hence, before constructing a join action, we require that

i. $A(bs_i) \cap A(bs_j) = \varnothing$ for all $i, j \in \{1, \ldots, n\}$ such that $i \neq j$.

Further, it is required that - before forming a partnership - multiple belief states representing different actors are not contradictory with respect to non-actor specific belief state variables. For instance, within our running example, joining two belief states with the non-actor-specific belief state variable *contract closed* being {*true*} in one of the belief states and being {*false*} in the other belief state would lead to a contradiction:

ii. For each $bs_i, i \in \{1, \ldots, n\}$: for each $(v(p), r(p), a(p)) \in bs_i$ with $a(p)$={*non-actor*}: for each $bs_j,\ j \neq i$: there is a $(v(p_j), r(p_j), a(p_j)) \in bs_j$ with $v(p_j) = v(p)$ such that $r(p_1) \cap \ldots \cap r(p_n) \cap r(p) \neq \varnothing$.

We further want to avoid the construction of unnecessary join actions. We thus require that the formed partnership is able to conduct at least one action. We ensure this with the following criterion iii. that has to be met by the joint belief state *bs* before constructing the according join action:

iii. In bs, at least one action is applicable (cf. Definition 2.3.11).

In a second step, when disbanding a partnership, we need to ensure that the resulting process model does not contain logical contradictions and thus a belief state *bs* with the partnership $A(bs)$ can be *split* into the belief states $bs_1,\ldots,bs_n$ with the partnerships $A(bs_1),\ldots,A(bs_n)$ by split actions only if the following criteria i. and ii. are fulfilled. These criteria are the counterparts to the previously defined criteria for forming a partnership. First, after disbanding a partnership, each actor may be contained in exactly one state after disbanding the partnership (cf. i.). This again results from the fact that the current status of an actor is always represented by one single belief state. Further, we need to ensure that each and every actor contained in the to-be-disbanded partnership is contained in a belief state after splitting the belief state (cf. ii.):

i. $A(bs_i) \cap A(bs_j) = \varnothing$ for all $i, j \in \{1, \ldots, n\}$ such that $i \neq j$

ii. $\cup_i A(bs_i) = A(bs)$

Additionally, we again ensure that at least one action is applicable in each belief state after splitting to avoid the construction of unnecessary split actions (cf. iii.). This, together with ii., is required as otherwise, actors would possibly not be able to reach their individual goal state(s):

iii. In each belief state bs_i, at least one action is applicable (cf. Definition 2.3.11).

For each set of belief states that meets the criteria for being joined, respective each single belief state that meets the criteria for being split, we construct the according join actions resp. split actions by means of an algorithm, which is presented in the following subsection.

2.3.4.4 Algorithm

Existing single-actor planning approaches (e.g., Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2012; Heinrich et al., 2015c; Henneberger et al., 2008; Hoffmann et al., 2009) construct planning graphs by means of a forward search that iteratively ***1***) retrieves which actions are applicable in a belief state and ***2***) generates the next belief state for each of these actions. We adopt these approaches, consisting of the major phases *identification of applicable actions* and *retrieval of next belief state*, but extend them for planning multi-actor process models (cf. contribution ❷). Our algorithm is presented in form of a pseudocode (see Appendix 7.3.1) and outlined in a textual description.

The algorithm works iteratively, starting with the initial belief states. For a belief state, it ***1a***) checks which actions are applicable (cf. Definition 2.3.11; line 4 of Listing 7.16 in Appendix 7.3.1) in the considered belief state. Further, actions that ***1b***) can be conducted by a subset of the actors represented in the belief state (line 6 of Listing 7.16) and

actions that ***1c***) can possibly be conducted by a partnership that needs to be formed (line 8 of Listing 7.16) are identified.

For each action identified as applicable (cf. step *1a*)), ***2a***) the belief state resulting from the application of the action is constructed and planned (line 5 of Listing 7.16). If an action can be conducted by a subset of the actors (cf. step *1b*); line 6 of Listing 7.16 and SUB disband; cf. Listing 7.19), ***2b***) according split actions and the subsequent belief states are constructed automatically (line 10 in Listing 7.19), based on the belief state and the information which (smaller partnership of) actor(s) could conduct the action. If a partnership can possibly be formed to conduct the action (cf. step *1c*); line 8 in Listing 7.16 and SUB join; cf. Listing 7.18), ***2c***) the action together with the currently considered preceding belief state is saved as *potentially suitable for cooperation* (line 2 in Listing 7.18).

Further, in step ***2d***) such an identified possibility for cooperation is matched with other combinations of belief states with the considered action already identified in preceding iterations (line 3 in Listing 7.18). If thereby an action is identified as applicable by a partnership of actors represented in different belief states (and thus all criteria i. to iii. are fulfilled), ***2e***) the algorithm subsequently performs an automated planning of join actions (line 7 in Listing 7.18). These join actions create a joint belief state by means of a regular state transition. They thus enable a joint conduction of the action (in the joint belief state) by actors that formerly were represented in their own individual belief states or cooperated in smaller partnerships. After ***2e***), the next iteration step starts.

To sum up the proposed approach and to illustrate the resulting planning graph, Figure 2.10 shows an excerpt of our running example. Each conducting actor starts with an individual initial state (cf. Definition 2.3.9), denoted by means of a square, tagged with IS and the according name of the actor, at the leftmost area in Figure 2.10. The actors need to form a partnership in order to conduct the action *job interview* jointly. This is achieved by join actions labelled with 'cooperate' (actions are denoted by rounded rectangles). The partnership is represented by the joint belief state (tabular representation of belief state tuples) in the left area of the detailed excerpt framed by the dashed line. Then, the action *job interview* – our focus in Figure 2.10 – is planned for *Bob*, *Danielle* and *Eric* (here, the not mandatory chair member *Silvia* participates as well). It leads to the following belief state at the right of the detailed excerpt. Subsequently, the applicant will be engaged or rejected (see actions at the top area of Figure 2.10).

2.3.5 Evaluation

In order to provide a 'proof of the solution' (Bertrand & Fransoo, 2002), we evaluated the validity (E1) of our approach. Furthermore, as proposed by Meredith et al. (1989),

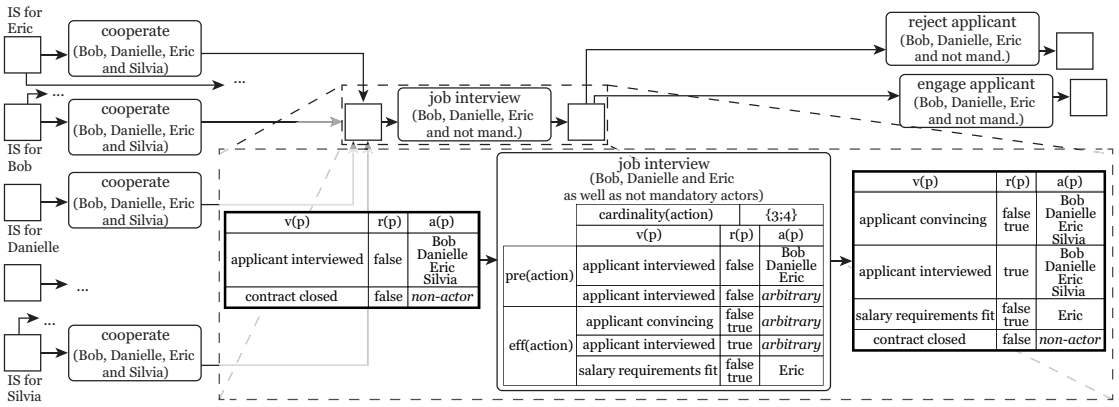

Figure 2.10: Excerpt of the Planning Graph of the Running Example

we evaluated the technical and practical feasibility (E2) as well as the effectiveness of our approach (E3) by means of a prototypical implementation and its application in real-world scenarios.

2.3.5.1 Assessment of the Validity (E1)

To assess the validity (E1), we conducted a mathematical evaluation of our approach by proving the key properties termination, correctness (i.e., all planned paths are feasible) and completeness (i.e., all feasible paths from an initial state to a goal state are planned). Due to length restrictions, we refer to Appendix 7.3.2 for the proofs. The proofs show that our algorithm terminates and the multi-actor process models constructed by our approach in an automated manner are indeed correct and complete (cf. contribution ❷).

2.3.5.2 Assessment of the Technical and Practical Feasibility (E2)

When evaluating the technical and practical feasibility of our approach (E2), we examined these criteria regarding the algorithm and the underlying conceptual foundation by analysing the following three evaluation questions:

(E2.1) *Can the approach be instantiated in a prototypical implementation?*

(E2.2) *Is it possible to apply the approach to real-world scenarios and how can the necessary input data (i.e., the specification of actors, actions, initial states and conditions for goal states) be obtained?*

(E2.3) *What are the results of these applications in terms of correctness of the constructed multi-actor process models? What are the key properties of these models and how long does their automated planning take?*

With respect to (E2.1), a Java implementation of a single-actor process planning algorithm (cf. Bertoli et al., 2006; Heinrich & Schön, 2015) served as a basis for our work. We extended this implementation to incorporate the presented algorithm that enables the automated construction of multi-actor process models (see Appendix 7.3.1 for the pseudocode of the algorithm). Actors, actions, initial states and goal states can be imported into the prototype by means of XML files. We ensured the validity of our prototype by means of structured tests using the JUnit framework. Here we carried out extreme value tests, unit tests and regression tests (i.e., validation that single-actor process models could still be planned correctly). Further, persons other than the programmers validated the source code via a structured walkthrough. At the end of the test phase, the implementation did not show any errors, supporting the technical feasibility of our approach and providing "proof by construction".

In regard to (E2.2) we analyzed whether it is possible to apply the approach to real-world scenarios (i.e., the scenarios *Human Resources*, *Product Manufacturing*, *Healthcare* and five further scenarios from the European financial services providers discussed in the introduction) using our prototypical implementation. In particular, we analyzed whether and in which way it is possible to obtain the necessary input data to apply the approach. Our study showed that the necessary input data could be obtained in different ways. On the one hand, we, for instance, revised and extended existing single-actor planning specifications (i.e., input about participating actors such as employees or departments) to enable the planning of multi-actor process models. On the other hand, we have been able to formalize the informal information provided by domain experts so that our approach could be applied. This is of particular interest for process modeling projects in practice where domain experts and business analysts often closely work together to construct process models. In Table 2.4 we give details about how the necessary input data was obtained, similar to Siha and Saad (2008). Due to length restrictions, we concentrate on the scenarios *Human Resources*, *Product Manufacturing* and *Healthcare*. However, we also applied our approach to five further scenarios from European financial services providers, where the data provided by these companies could successfully be used as input data (if desired, we can provide much further details for these five scenarios as well).

In regard to (E2.3), we applied our approach to these scenarios and aimed at evaluating in how far our approach is suitable for providing a conceptual foundation (cf. contribution ❶) for multi-actor process models in real-world scenarios and to which extent the results of the approach correspond to the actually conducted processes in the scenarios. Thus, we evaluated in detail whether all Aspects (A1) to (A3) of contribution ❶ are appropriately taken into account in the resulting process models. Precisely, we evaluated whether (A1) all individual starting points and goals of conducting actors,

Scenario	Human Resources	Product Manufacturing	Healthcare
Description of the scenario	Hiring at a university, starting from an application up to the processing of the hiring decision	Handling of incoming orders at a manufacturing company up to the shipping of the ordered products	Basic course of surgery in a hospital, starting from an preliminary talk up to the patient leaving the hospital
Context of the scenario	Real process, conducted at a chair of a university	Real process as conducted at a manufacturing company for electrical devices	Simplified version of real process as conducted at a hospital
Context of the data provisioning	Experimental case study with chair; aim of project: visualisation/documentation of conducted process	Experimental case study with the head of production department; aim of project: support of process improvement; three interviews	Experimental case study with an experienced intensive care surgical nurse of a hospital; aim: evaluation of the presented approach; two interviews
Basis of provided information	Existing scenario specified (in terms of XML files) for single-actor planning algorithm	Existing scenario specified (in terms of XML files) for single-actor planning algorithm	Informal information provided by intensive care surgical nurse
Formalization and refinement of provided information	Retrieval in two steps: *First step:* ● Determination of responsible actors from action names of single-actor specification (e.g., *research project manager checks CV actor research project manager*) ● Retrieval of actions and initial states / goal states based on single-actor specification *Second step:* ● Revision of actions regarding required cooperation of the conducting actors based on transcripts of an interview with according employees (e.g., the definition of the action *job interview* was extended as it, according the transcripts, has to be conducted jointly by multiple actors) ● Preparation of XML specifications	Retrieval in two steps:*First step:*Determination of responsible actors from action names of single-actor specification (e.g., *warehouse checks stock actor warehouse*) Retrieval of actions and initial states / goal states based on single-actor specification *Second step:*Revision of actions regarding required cooperation of the conducting actors based on interviews with the head of production department (e.g.: the definition of the action *hand over shipment notification* was extended as it, according the head of production department, has to be conducted jointly by the sales department and the warehouse department) Preparation of XML specifications	Retrieval of formal and informal information based on two interviews:*First interview:*Required staff for a typical surgery (participating actors) His tasks during the surgery *Second interview* (*after surgical nurse consulted colleagues*):Actions conducted jointly with other actors Individual initial and goal states of all actors; (e.g., the surgeon is no longer needed – formally: his/her goal state is met - as soon as the patient is transferred to recovery; ward nurse's goal is met when patient is in his/her room again) *After second interview:*Refinement and formalization of the provided informal information in terms of XML specifications
Actors involved	Applicant / two research project managers / personnel officer / chair member / gender representative	Warehouse department / production department / sales department	Surgeon / anaesthesiologist / surgical nurse / ward nurse / patient

Table 2.4: Evaluation of our Approach with regard to (E2.2)

(A2) all partnerships conducting actions jointly as well as (A3) all required join actions and split actions were appropriately contained in the real-world scenarios. Similar to the presentation in Siha and Saad (2008), we discuss the evaluation of our approach with regard to (E2.3) in Table 2.5, where we again focus on the three scenarios *Human Resources*, *Product Manufacturing* and *Healthcare* (comparable findings could be provided for the other evaluated scenarios as well).

In the application to these real-world scenarios, all necessary individual starting points and goals of actors as well as partnerships conducting actions jointly were represented and all join and split actions were constructed correctly according to the provided input. We evaluated this by a structured walkthrough of the constructed process models and by examining whether each applicable action as well as each necessary join and split action was planned and whether all planned actions were correct and actually necessary.

We further examined the key properties of the multi-actor processes and the according multi-actor process models resulting from applying our approach. As seen in Table 2.6, we first determined the number of actors conducting the processes as well as the number of belief states, join actions, split actions, actions conducted in a partnership and actions in total in the multi-actor process models. Additionally, we identified the number of partnerships as well as the minimum and maximum number of actors cooperating in a partnership for each process. Lastly, we determined the required runtime for planning the multi-actor process models (executed on an Intel Core i7-2640M, 2.80 GHz, Windows 8.1 64 bit, Kernel Version 6.3.9600, Java 8). The process models are of small to large size, containing between 20 and 212 actions in total. This is also reflected by the number of actors, which ranges from two to eight actors that form a maximum of up to seven different partnerships. These partnerships conduct between four and 19 actions throughout the respective processes and consist of two up to four actors. Our approach was capable of constructing the multi-actor process models regardless of their size and complexity. Overall, the required runtime for planning multi-actor process models comprising a significant number of actors, partnerships as well as join and split actions still was below four seconds, which supports the practical feasibility of our approach.

To sum up, our approach was prototypically implemented, provided a suitable conceptual foundation for the resulting small, medium-sized and large multi-actor process models in several real-world scenarios and their automated planning could be completed in appropriate time. These results support the technical and practical feasibility of our approach.

Scenario	Human Resources	Product Manufacturing	Healthcare
(A1) all individual starting points and goals of conducting actors are appropriately contained	Applicant, gender representative, personnel officer and chair member start from individual actor-specific initial states and possess individual actor-specific goal states; research project managers start at common actor-specific initial state and possess a common actor-specific goal state; example: process of personnel officer starts with incoming application and ends when information about applicant being hired or not is sent. **All six actors as well as their according initial states and goal states are correctly included in the resulting process model.**	Each of the three actors starts from an actor-specific initial state and possesses at least one actor-specific goal state; example: process of sales department starts with an incoming order and ends when the order is declined or shipment information has been sent to customer. **All three participating departments (represented by actors) as well as their according initial states and goal states are correctly included in the resulting multi-actor process model.**	Each of the five actors starts from an individual actor-specific initial state and possesses an actor-specific goal state; example: process of ward nurse starts with the patient being in in his/her room before transferring him/her to anteroom of surgery and ends when the patient is back in his/her room after the surgery. **All five actors as well as their according initial states and goal states are correctly included in the resulting process model.**
(A2) all partnerships conducting actions jointly are appropriately contained	During the process, different partnerships are needed to conduct actions; example: both research project managers as well as (not mandatory) chair member check application documents. **15 actions are conducted jointly during the process; four distinct partnerships (consisting of up to four actors) that conduct these actions are correctly included in the resulting process model.**	Partnerships consisting of particular actors are needed to conduct four actions; example: warehouse department and production department have to jointly conduct create production requirements; based on ordered amount, available warehouse stock and already dispatched production output, amount of required additional production output is retrieved. **Four actions are conducted in respective partnerships (max size: two actors); all required partnerships are correctly included in the resulting process model.**	During the process, the actors need to form different partnerships; example: surgeon and patient conduct a common preliminary talk. **13 actions are conducted in respective partnerships (max size: four actors); all required partnerships are correctly included in the resulting process model.**
(A3) all required join actions and split actions are appropriately contained	To form resp. disband the partnerships required for the process (cf. Aspect (A2)), join actions resp. split actions are constructed by our approach in an automated manner. For instance, three join actions form the partnership that allows conducting the action check application documents. Two of these join actions incorporate the cooperation of the respective research project managers with each other as well as the chair member while the third join action incorporates the cooperation of the chair member with both research project managers. **The resulting multi-actor process model contains all 23 necessary join actions and all 23 necessary split actions to enable a correct conduction of the process.**	To form resp. disband the partnerships required for the process (cf. Aspect (A2)), join actions resp. split actions are constructed by our approach in an automated manner. For instance, two join actions form the partnership that allows conducting the action create production requirements. One of the join actions incorporates the cooperation of the warehouse department with the production department and the other join action inversely incorporates the cooperation of the production department with the warehouse department. **The resulting multi-actor process model contains all eight necessary join actions and all six necessary split actions to enable a correct conduction of the process.**	To form resp. disband the partnerships required for the process (cf. Aspect (A2)), join actions resp. split actions are constructed by our approach in an automated manner. For instance, two join actions form the partnership that allows conducting the action preliminary talk. One of the join actions incorporates the cooperation of the patient with the surgeon and the other join action inversely incorporates the cooperation of the surgeon with the patient. **The resulting multi-actor process model contains all 156 necessary join actions and all 24 necessary split actions to enable a correct conduction of the process.**

Table 2.5: Evaluation of our Approach with regard to (A1)-(A3)

Scenario context	University	Manufacturing company	German regional hospital	European insurance company			European bank	
Scenario name	**Human Resources**	**Product Manufacturing**	**Healthcare**	**Allocation of project resources**	**Preparation of decision template for project application**	**Project closure**	**Transfer of investment deposit**	**Combination of multiple investment accounts**
Number of actors in process model	6	3	5	8	2	7	7	6
Number of belief states in process model	67	27	126	76	15	43	24	35
Number of join actions in process model	23	8	156	30	6	23	9	18
Number of split actions in process model	23	6	24	30	6	14	3	10
Number of actions conducted in a partnership in process model	15	4	13	19	4	11	11	9
Number of actions (in total) in process model	75	26	212	92	20	51	25	42
Number of partnerships (min. size / max. size) in process model	4 (2/4)	2 (2/2)	6 (2/4)	7 (2/3)	1 (2/2)	6 (2/3)	4 (2/4)	5 (2/2)
Runtime for planning the multi-actor process model	3.427 s	0.021 s	0.161 s	0.128 s	0.004 s	0.032 s	0.032 s	0.014 s

Table 2.6: Key Properties of the constructed Multi-Actor Process Models and Runtime for planning them

2.3.5.3 Assessment of Effectiveness (E3)

In order to assess the effectiveness of our approach, we discuss the following evaluation question:

(*E3*) *Are the constructed multi-actor process models feasible according to the assessment of a practitioner in an experimental setting?*

To evaluate the effectiveness in real-world scenarios, we applied our approach in an experiment (cf. Meredith et al., 1989). Due to length restrictions, we focus on the real-world scenario *Healthcare* by examining a surgery process and present our findings within this scenario.

The environment of this experiment as well as its results are presented in Table 2.7. In this setting, we aimed to evaluate whether our approach constructs feasible multi-actor process models that appropriately reflect processes as conducted in reality in regard to the assessment of a practitioner. Similar to process modeling in a real business environment, an experienced intensive care surgical nurse (domain expert) provided us with detailed information about the basic course of a surgery and the involved actors in two interviews (cf. also the corresponding description in Table 2.4). As the surgical nurse was not familiar with process modeling we refined and formalized the informal information he gave us and hence specified the actors, preconditions, cardinalities and effects of actions in terms of the aforementioned XML files. We thereafter were able to plan a multi-actor process model that comprised 156 join actions and 24 split actions (see Table 2.6) by means of our prototypical implementation.

We then asked him whether the constructed multi-actor process model appropriately reflects the starting point and goals of the surgical nurse in the process (Aspect (A1)), partnerships including the surgical nurse conducting actions jointly (Aspect (A2)) as well as the join actions and split actions in which the surgical nurse participates (Aspect (A3)). Table 2.7 describes the assessment of the surgical nurse regarding Aspects (A1) to (A3) in detail (structured in a similar way as Siha et al. (2008) present their findings).

To sum up, the experimental evaluation together with a practitioner supported the effectiveness of our approach to construct feasible multi-actor process models since an actor-specific initial state and actor-specific goal states, partnerships as well as join actions and split actions were considered as valid by the practitioner.

To conclude, the analysis of the evaluation questions supports the validity, the technical and practical feasibility and the effectiveness of the presented approach. Table 2.8 summarizes the results.

Scenario	**Healthcare**
Description of the scenario	Basic course of surgery in German hospital
Way of assessing the model	*Further* (*third*) *interview*: Step-by-step discussion of the model with the surgical nurse, focusing (primarily, not exclusively) on the actions he has to perform (walkthrough); brief description of the model so that he could understand it (as he was not familiar with process modeling notations); verbal discussion of Aspects (A1) to (A3); focus on the sequence of actions/tasks as well as the partnerships he joined throughout the process
Results (according to the assessment of the surgical nurse)	
(A1) individual starting points and goals of conducting actors correspond to those in the actually conducted process	We described the meaning of the belief state variables contained in the initial states and goal states. For instance: Discussion of the belief state tuples of the initial state in which his process starts; surgical nurse stated that he, as correctly represented in the initial state, starts in regular clothes in the anteroom. He further stated that his process ends when the paperwork is done after the actual surgery, which is also correctly represented in the according goal state. **The surgical nurse confirmed that, in his view, initial state and goal state in the multi-actor process model accurately reflect the respective states compared to the conduction of the process in reality.**
(A2) partnerships conducting actions jointly correspond to those in the actually conducted process	Discussion of the partnerships of the multi-actor process model in which the surgical nurse participates according to the model; in particular: clarification whether he actually participates in these partnerships in a real surgery. He agreed that, for instance, he brings the patient to and from the surgery room (formally: joins a partnership with the patient) and finishes the paperwork without any actor; he further stated that – at least spontaneously – he could not think of a partnership occurring in reality but not represented in the process model. **The surgical nurse confirmed that, in his view, partnerships are appropriately contained in the multi-actor process model and reflect the partnerships as formed during the process in reality.**
(A3) join actions and split actions correspond to those in the actually conducted process	Additional clarification about the meaning of the split actions was necessary; we elaborated that they tell an actor to "leave a partnership" and to continue with his/her individual tasks or with joining a different partnership with other actors; thereafter, he confirmed that, for instance, the partnership conducting the surgery is correctly split; anesthesiologist and surgeon leave the surgery room. **The surgical nurse confirmed that, in his view, join and split actions are appropriately contained in the multi-actor process model and reflect the respective actions during the process in reality.**

Table 2.7: Evaluation of our Approach with regard to (E3)

Evaluation Question	Result
(E1) Does the approach terminate and provide correct and complete multi-actor process models?	A mathematical evaluation of the approach proves that these criteria hold.
(E2.1) Can the algorithm be instantiated in a prototypical implementation?	The algorithm was implemented and successfully integrated into a prototype for the automated planning of process models.
(E2.2) Is it possible to apply the algorithm to real-world scenarios and how can the necessary input data (i.e., the specification of actors, actions, initial states and conditions for goal states) be obtained?	The algorithm was applied to several real-world scenarios. The necessary input data could, for instance, be obtained by analysing and refining existing specifications for single-actor process models or by interviewing a participant of the process and formalizing the provided data in terms of XML files.
(E2.3) What are the results of these applications in terms of correctness of the constructed multi-actor process models? What are the key properties of the constructed multi-actor process models and how long does it take to construct these models?	Multi-actor process models were constructed for each of the real-world scenarios. The Aspects (A1) to (A3) were fulfilled in each case. The constructed multi-actor process models have been of small to large size (regarding the number of actions, conducting actors and partnerships). The runtime for planning such multi-actor process models comprising a significant number of join and split actions was below four seconds.
(E3) Are the constructed multi-actor process models feasible according to the assessment of a practitioner in an experimental setting?	The practitioner confirmed that (A1) initial states and goal states of the multi-actor process model reflected the respective states in reality; (A2) all partnerships contained in the multi-actor process model constructed by the approach corresponded to those formed in the actually conducted process; (A3) the join and split actions contained in the multi-actor process model as well as the model itself were feasible.

Table 2.8: Results with regard to all Evaluation Questions

2.3.6 Conclusion, Limitations, and Future Work

In this paper, we presented an approach for the automated planning of multi-actor process models (cf. contribution ❷) based on a conceptual foundation (cf. contribution ❶). We described how to extend a common planning domain in the literature to enable taking actor-specific information and individual starting points and goals into account (Aspect (A1)). Our approach can further cope with cardinalities of partnerships (i.e., sets of actors) required to conduct an action (Aspect (A2)). Moreover, we outlined how to construct join and split actions in an automated manner. These actions incorporate the cooperation of multiple actors in the control flow of process models and hence support individual actors by determining explicitly at which steps in a process they can or need to cooperate in partnerships to achieve their individual goals (Aspect (A3)). As our approach extends existing single-actor planning approaches, compatibility with prevalent works is supported. Our approach is evaluated by means of mathematical proofs of its key properties, a prototypical implementation, the application to real-world scenarios, a detailed analysis of the constructed multi-actor process models regarding Aspects (A1) to (A3), runtime analyses and the assessment of a practitioner in an experimental real-world scenario.

Our work addresses an important subproblem of the research field automated planning of process models, namely the automated planning of multi-actor process models. This issue has not been addressed so far and hence we believe that our work significantly increases the scope of that research field. Furthermore, it contributes to the general research field of business process modeling by presenting a new approach to represent multi-actor processes that comprise partnerships conducting parts of processes jointly. Existing modeling approaches and notations such as swimlanes have several shortcomings, resulting in 'messy and difficult to understand' process models (Pulgar et al., 2017). Hence, we include the cooperation of actors in the control flow of process models by constructing explicit actions determining where in the process to form and to disband partnerships. Additionally, we address a relevant problem in practice as multi-actor process models are widespread in today's business world. For instance, we strongly supported the analysis of about 600 core processes of two European financial services providers. In this context, over 60% of the analyzed processes of the insurance company comprise partnerships of three or more actors. The proposed approach enables practitioners to represent multi-actor process models and to denote actions that have to be performed by a partnership of multiple actors in contrast to existing approaches. Lastly, as runtimes for the automated planning of multi-actor process models were short, the proposed approach enables companies to construct multi-actor process models in appropriate time and thus to stay flexible and competitive.

However, there are some limitations of our work which have to be addressed in future research. First, to increase the acceptance of our approach in an industrial setting and hence to enable process modelers or even domain experts without expertise in process modeling to construct multi-actor process models, the prototypical implementation needs to be extended in terms of a graphical user interface. There exists a graphical user interface for the single-actor process planning approach that served as a basis for our prototypical implementation. This enables modelers to specify actions, including preconditions and effects, as well as initial and goal states of single-actor processes. However, this graphical user interface needs to be extended to allow the definition of multi-actor process models, comprising partnerships of actors, with individual initial states and goal states as well as actions that have to be conducted by these partnerships.

Second, process models can be hard to grasp for humans, especially when they represent complex processes (e.g., many actions and control flow structures) that are conducted by a large number of actors. Thus, future research should strive to alleviate this issue. A promising idea could be to provide an "actor-specific view" of the multi-actor process models constructed by our approach by focusing on and representing only actions and belief states that are relevant for the conduction of a specific actor.

Third, multi-actor processes in practice can vary considerably with regard to participating actors, size, goals and additional criteria. While the application of our approach in multiple real-world scenarios showed its feasibility and effectiveness, an application in further contexts could provide a more thorough verification of its practical feasibility.

Fourth, when applying the approach in real-world scenarios, "noisy" preconditions or effects of actions may occur (e.g., an interviewee is uncertain to specify starting from what order amount a control by three different actors is necessary regarding regulatory compliance) and influence the multi-actor process model resulting from planning. To address this issue, multiple plannings with different preconditions and/or effects of respective actions could be conducted. Based on this, it can be evaluated whether and to what extent (i.e., which actions) the resulting process model is influenced by the "noise" at all. This supports the determination of a feasible process model under such circumstances.

Fifth, in this paper we presented how preconditions and effects of actions can be specified on a per-actor basis. The processes we analyzed together with European financial services providers oftentimes contain actors representing departments consisting of multiple individuals. However, planning process models on the basis of individuals may sometimes be preferable. For instance, this may be beneficial in the case of actions that require a particular number of actors of a department. For such a more fine-grained planning, it would be promising to allow a role-based specification (as it is possible, for instance, in security related topics like access control) of preconditions and effects. Fol-

lowing this, a department consisting of multiple persons could be represented by a role and the individual persons could be specified by role-based preconditions and effects. In future, the presented approach can be enhanced to incorporate such role-based specifications by subsuming actors as well as action-specifications under roles and requiring a subset of the actors of each role for role-based preconditions.

2.3.7 References

Becker, J., Beverungen, D., Knackstedt, R., Matzner, M., Müller, O., and Pöppelbuß, J. (2013). "Designing Interaction Routines in Service Networks." In: *Scandinavian Journal of Information Systems* 25.1, pp. 37–68.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bertrand, J. W. M. and Fransoo, J. C. (2002). "Operations management research methodologies using quantitative modeling." In: *International Journal of Operations & Production Management* 22.2, pp. 241–264.

Boukhedouma, S., Alimazighi, Z., and Oussalah, M. (2017). "Adaptation and Evolution Frameworks for Service Based Inter-Organizational Workflows." In: *International Journal of E-Business Research* 13.2, pp. 28–57. ISSN: 1548-1131. DOI: 10.4018/IJEBR.2017040103.

Business Process Model and Notation (BPMN): Version 2.0.2 (2013). URL: http://www.omg.org/spec/BPMN/2.0.2/PDF (visited on 10/16/2014).

Cabanillas, C., Resinas, M., Mendling, J., and Ruiz-Cortés, A. (2015). "Automated team selection and compliance checking in business processes." In: *Proceedings of the 2015 International Conference on Software and System Process*, pp. 42–51.

Chen, Q. and Hsu, M. (2001). "Inter-enterprise collaborative business process management." In: *Proceedings*. Los Alamitos, Calif. [u.a.]: IEEE Computer Society, pp. 253–260. ISBN: 0-7695-1001-9.

Chinosi, M. and Trombetta, A. (2012). "BPMN: An introduction to the standard." In: *Computer Standards & Interfaces* 34.1, pp. 124–134. ISSN: 0920-5489.

Chiu, D., Karlapalem, K., and Li, Q. (2003). "Views for Inter-organization Workflow in an E-commerce Environment." In: *Semantic Issues in E-Commerce Systems*. Ed. by

R. Meersman, K. Aberer, and T. Dillon. Vol. 111. IFIP - The International Federation for Information Processing. Springer US, pp. 137–151. ISBN: 978-1-4757-1035-9. DOI: 10.1007/978-0-387-35658-7_9. URL: http://dx.doi.org/10.1007/978-0-387-35658-7_9.

Chouhan, S. S. and Niyogi, R. (2017). "MAPJA: Multi-agent planning with joint actions." In: *Applied Intelligence* 14.4, p. 105. DOI: 10.1007/s10489-017-0938-8.

Crosby, M., Rovatsos, M., and Petrick, R. P. A. (2013). "Automated Agent Decomposition for Classical Planning." In: *Proceedings of the 23rd International Conference on Automated Planning and Scheduling*.

Davenport, T. H. and Short, J. (1990). *Information technology and business process redesign*. Taylor & Francis US.

De Weerdt, M. and Clement, B. (2009). "Introduction to planning in multiagent systems." In: *Multiagent and Grid Systems - Planning in multiagent systems* 5.4, pp. 345–355.

Dimopoulos, Y. and Moraitis, P. (2006). "Multi-Agent Coordination and Cooperation through Classical Planning." In: *2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pp. 398–402. DOI: 10.1109/IAT.2006.90.

Ding, Z., Sun, Y., Liu, J., Pan, M., and Liu, J. (2015). "A genetic algorithm based approach to transactional and QoS-aware service selection." In: *Enterprise Information Systems*, pp. 1–20. ISSN: 1751-7575. DOI: 10.1080/17517575.2015.1048832.

Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). "A survey of autonomic communications." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 1.2, pp. 223–259. ISSN: 1556-4665.

Ephrati, E. and Rosenschein, J. S. (Aug. 1994). "Divide and conquer in multi-agent planning." In: *Proceedings of the 12th National Conference on Artificial Intelligence*. Vol. 1. Seattle, WA, USA, p. 80.

Fahland, D., Favre, C., Koehler, J., Lohmann, N., Völzer, H., and Wolf, K. (2011). "Analysis on demand: Instantaneous soundness checking of industrial business process models." In: *Data & Knowledge Engineering* 70.5, pp. 448–466. DOI: 10.1016/j.datak.2011.01.004.

Falou, M. E., Bouzid, M., Mouaddib, A.-I., and Vidal, T. (2009). "Automated Web Service Composition: A Decentralised Multi-agent Approach." In: *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pp. 387–394. DOI: 10.1109/WI-IAT.2009.68.

Fikes, R. E. and Nilsson, N. J. (1971). "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving." In: *Proceedings of the 2Nd International Joint Conference on Artificial Intelligence*. IJCAI'71. San Francisco, CA, USA: Morgan Kaufmann

Publishers Inc, pp. 608–620. URL: http://dl.acm.org/citation.cfm?id=1622876.1622939.

Fleischmann, A., Kannengiesser, U., Schmidt, W., and Stary, C. (2013). "Subject-Oriented Modeling and Execution of Multi-agent Business Processes." In: *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*. IEEE, pp. 138–145. ISBN: 978-0-7695-5145-6. DOI: 10.1109/WI-IAT.2013.102.

Forstner, E., Kamprath, N., and Röglinger, M. (2014). "Capability development with process maturity models – Decision framework and economic analysis." In: *Journal of Decision Systems* 23.2, pp. 127–150. ISSN: 1246-0125. DOI: 10.1080/12460125.2014.865310.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Ghallab, M., Nau, D. S., and Traverso, P. (2016). *Automated Planning and Acting*. New York, NY: Cambridge University Press. ISBN: 978-1107037274.

Ghrab, S., Saad, I., Kassel, G., and Gargouri, F. (2017). "A Core Ontology of Know-How and Knowing-That for improving knowledge sharing and decision making in the digital age." In: *Journal of Decision Systems* 26.2, pp. 138–151. ISSN: 1246-0125.

Grefen, P., Aberer, K., Hoffner, Y., and Ludwig, H. (2000). "CrossFlow: cross-organizational workflow management in dynamic virtual enterprises." In: *International Journal of Computer Systems Science & Engineering* 15.5, pp. 277–290.

Havur, G., Cabanillas, C., Mendling, J., and Polleres, A. (2015). "Automated Resource Allocation in Business Processes with Answer Set Programming." In: *11th International Workshop on Business Process Intelligence*.

Havur, G., Cabanillas, C., Mendling, J., and Polleres, A. (2016). "Resource Allocation with Dependencies in Business Process Management Systems." In: *Business Process Management Forum: BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings*. Ed. by M. La Rosa, P. Loos, and O. Pastor. Cham: Springer International Publishing, pp. 3–19. ISBN: 978-3-319-45468-9. DOI: 10.1007/978-3-319-45468-9_1.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems (ICIS)*. Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., Lewerenz, L., and Zimmermann, S. (Mar. 2015a). "Quality of Service-aware Service Selection: A Novel Approach Considering Potential Service Failures and Non-Deterministic Service Values." In: *INFORMS Service Science, A Jour-*

nal of the Institute for Operations Research and the Management Sciences 7.1, pp. 48–69. DOI: 10.1287/serv.2015.0093.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Heinrich, B. and Schön, D. (2015c). "Automated Planning of context-aware Process Models." In: *Proceedings of the 23rd European Conference on Information Systems (ECIS)*. Ed. by J. Becker, J. vom Brocke, and M. de Marco. Münster, Germany, Paper 75.

Heinrich, B. and Schön, D. (2016). "Automated Planning of Process Models: The Construction of Simple Merges." In: *Proceedings of the 24rd European Conference on Information Systems (ECIS)*. Istanbul, Turkey.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik (MKWI)*. Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hoffmann, J., Weber, I., and Kraft, F. M. (2009). "Planning@ sap: An application in business process management." In: *Proceedings of the 2nd International Scheduling and Planning Applications woRKshop (SPARK'09)*.

Hoffmann, J., Weber, I., and Kraft, F. M. (July 2012). "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management." In: *Journal of Artificial Intelligence Research* 44.1, pp. 587–632. DOI: 10.1613/jair.3636.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference on Advanced Information Systems Engineering (CAiSE)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

Huang, J. S., Hsueh, K., and Reynolds, A. (2013). "A framework for collaborative social, economic and environmental development: Building a digital ecosystem for societal empowerment." In: *7th IEEE International Conference on Digital Ecosystems and Technologies (DEST) - Complex Environment Engineering*, pp. 166–171. DOI: 10.1109/DEST.2013.6611348.

IEEE Task Force on Process Mining (2012). "Process Mining Manifesto." In: *Business Process Management Workshops*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rose-

mann, M. J. Shaw, C. Szyperski, F. Daniel, K. Barkaoui, and S. Dustdar. Vol. 99. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 169–194. ISBN: 978-3-642-28107-5. DOI: 10.1007/978-3-642-28108-2_19.

Jennings, N. R., Norman, T. J., Faratin, P., O'Brien, P., and Odgers, B. (2000). "Autonomous agents for business process management." In: *Applied Artificial Intelligence* 14.2, pp. 145–189. ISSN: 0883-9514. DOI: 10.1080/088395100117106.

Kannengiesser, U. (2017). "The Future: Obstacles and Opportunities." In: *S-BPM in the Production Industry: A Stakeholder Approach*. Ed. by M. Neubauer and C. Stary. Cham: Springer International Publishing, pp. 209–230. ISBN: 978-3-319-48466-2. DOI: 10.1007/978-3-319-48466-2_8.

Katzmarzik, A., Henneberger, M., and Buhl, H. U. (2012). "Interdependencies between automation and sourcing of business processes." In: *Journal of Decision Systems* 21.4, pp. 331–352. ISSN: 1246-0125. DOI: 10.1080/12460125.2012.749755.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security* (*IJCSIS*) 7.2, pp. 147–152.

Klier, J., Klier, M., Müller, A.-L., and Rauch, C. (2016). "The impact of self-service technologies – towards an economic decision model and its application at the German Federal Employment Agency." In: *Journal of Decision Systems* 25.2, pp. 151–172. ISSN: 1246-0125. DOI: 10.1080/12460125.2016.1141274.

Kossak, F., Illibauer, C., Geist, V., Natschläger, C., Ziebermayr, T., Freudenthaler, B., Kopetzky, T., and Schewe, K.-D. (2016). "A Layered Approach for Actor Modelling." In: *Hagenberg Business Process Modelling Method*. Ed. by F. Kossak, C. Illibauer, V. Geist, C. Natschläger, T. Ziebermayr, B. Freudenthaler, T. Kopetzky, and K.-D. Schewe. Cham: Springer International Publishing, pp. 63–84. ISBN: 978-3-319-30495-3. DOI: 10.1007/978-3-319-30496-0_3.

Lambert, D. M., Emmelhainz, M. A., and Gardner, J. T. (1996). "Developing and Implementing Supply Chain Partnerships." In: *The International Journal of Logistics Management* 7.2, pp. 1–17.

Leymann, F., Roller, D., and Schmidt, M. T. (2002). "Web services and business process management." In: *IBM Systems Journal* 41.2, pp. 198–211.

Liu, C., Zeng, Q., Duan, H., and Lu, F. (2015). "Petri Net Based Behavior Description of Cross-Organization Workflow with Synchronous Interaction Pattern." In: *Process-Aware Systems*. Ed. by J. Cao, L. Wen, and X. Liu. Vol. 495. Communications in Computer and Information Science. Springer Berlin Heidelberg, pp. 1–10. ISBN: 978-3-662-46169-3. DOI: 10.1007/978-3-662-46170-9_1.

Liu, C. and Zhang, F. (2016). "Petri Net Based Modeling and Correctness Verification of Collaborative Emergency Response Processes." In: *Cybernetics and Information Technologies* 16.3. ISSN: 1314-4081. DOI: 10.1515/cait-2016-0038.

Mendling, J., Verbeek, H. M. W., van Dongen, B. F., van der Aalst, W. M. P., and Neumann, G. (2008). "Detection and prediction of errors in EPCs of the SAP reference model." In: *Data & Knowledge Engineering* 64.1, pp. 312–329. DOI: 10.1016/j.datak.2007.06.019.

Meredith, J. R., Raturi, A., Amoako-Gyampah, K., and Kaplan, B. (1989). "Alternative research paradigms in operations." In: *Journal of operations management* 8.4, pp. 297–326. ISSN: 0272-6963.

Meyer, H. and Weske, M. (2006). "Automated service composition using heuristic search." In: *Business Process Management*, pp. 81–96.

Mitroff, I. I., Betz, F., Pondy, L. R., and Sagasti, F. (1974). "On Managing Science in the Systems Age: Two Schemas for the Study of Science as a Whole Systems Phenomenon." In: *Interfaces* 4.3, pp. 46–58. ISSN: 0092-2102. DOI: 10.1287/inte.4.3.46.

Natschläger, C. and Geist, V. (2013). "A layered approach for actor modelling in business processes." In: *Business Process Management Journal* 19.6, pp. 917–932. ISSN: 1463-7154. DOI: 10.1108/BPMJ-10-2012-0107.

Nissim, R., Brafman, R. I., and Domshlak, C. (2010). "A general, fully distributed multi-agent planning algorithm." In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*. Ed. by M. Luck. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, pp. 1323–1330. ISBN: 978-0-9826571-1-9.

OMG Unified Modeling Language TM (OMG UML): Version 2.5 (2015). URL: http://www.omg.org/spec/UML/2.5 (visited on 05/04/2016).

Paik, I., Chen, W., and Huhns, M. N. (2014). "A scalable architecture for automatic service composition." In: *Services Computing, IEEE Transactions on* 7.1, pp. 82–95.

Peleteiro, A., Burguillo, J. C., Arcos, J. L., and Rodriguez-Aguilar, J. A. (2014). "Fostering Cooperation Through Dynamic Coalition Formation and Partner Switching." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 9.1, 1:1–1:31. ISSN: 1556-4665. DOI: 10.1145/2567928. URL: http://doi.acm.org/10.1145/2567928.

Pulgar, J. and Bastarrica, M. C. (2017). "Transforming Multi-role Activities in Software Processes into Business Processes." In: *Business Process Management Workshops: BPM 2016 International Workshops, Rio de Janeiro, Brazil, September 19, 2016, Revised Papers*. Ed. by M. Dumas and M. Fantinato. Cham: Springer International Publishing, pp. 372–383. ISBN: 978-3-319-58457-7. DOI: 10.1007/978-3-319-58457-7_27. URL: http://dx.doi.org/10.1007/978-3-319-58457-7_27.

Recker, J. C., Indulska, M., Rosemann, M., and Green, P. (2006). "How Good is BPMN Really? Insights from Theory and Practice." In: *Proceedings of the 14th European Conference on Information Systems* (*ECIS 2006*). Goeteborg, Sweden, Paper 135.

Rosemann, M. and vom Brocke, J. (2015). "The Six Core Elements of Business Process Management." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Heidelberg Dordrecht London New York: Springer, pp. 107–122.

Roy, S., Sajeev, A. S. M., Bihary, S., and Ranjan, A. (2014). "An Empirical Study of Error Patterns in Industrial Business Process Models." In: *IEEE Transactions on Services Computing* 7.2, pp. 140–153. ISSN: 1939-1374. DOI: 10.1109/TSC.2013.10.

Rozinat, A., Zickler, S., Veloso, M., van der Aalst, W. M. P., and McMillen, C. (2009). "Analyzing Multi-agent Activity Logs Using Process Mining Techniques." In: *Distributed Autonomous Robotic Systems 8*. Ed. by H. Asama, H. Kurokawa, J. Ota, and K. Sekiyama. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 251–260. ISBN: 978-3-642-00643-2. DOI: 10.1007/978-3-642-00644-9_22.

Russell, N., van der Aalst, W. M. P., ter Hofstede, A. H. M., and Edmond, D. (2005). "Workflow Resource Patterns: Identification, Representation and Tool Support." In: *Advanced Information Systems Engineering*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, O. Pastor, and J. Falcão e Cunha. Vol. 3520. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 216–232. ISBN: 978-3-540-26095-0. DOI: 10.1007/11431855_16.

Schönig, S., Cabanillas, C., Jablonski, S., and Mendling, J. (2015). "Mining the organisational perspective in agile business processes." In: *International Conference on Enterprise, Business-Process and Information Systems Modeling*, pp. 37–52.

Serve, M., Yen, D. C., Wang, J.-.-C., and Lin, B. (2002). "B2B–enhanced supply chain process: toward building virtual enterprises." In: *Business Process Management Journal* 8.3, pp. 245–253. DOI: 10.1108/14637150210428952.

Shapiro, R., White, S. A., Bock, C., Palmer, N., zur Muehlen, M., Brambilla. Marco, and Gagné, D. (2012). *BPMN 2.0 handbook second edition: Methods, concepts, case studies and standards in business process management notation*. 2nd ed. Lighthouse Point, FLa.: Future Strategies. ISBN: 978-0-9849764-0-9.

Shoham, Y. and Tennenholtz, M. (1995). "On social laws for artificial agent societies: off-line design." In: *Artificial Intelligence* 73.1, pp. 231–252. ISSN: 0004-3702.

Siha, S. M. and Saad, G. H. (2008). "Business process improvement: Empirical assessment and extensions." In: *Business Process Management Journal* 14.6, pp. 778–802.

Skjoett–Larsen, T., Thernøe, C., and Andresen, C. (2003). "Supply chain collaboration." In: *International Journal of Physical Distribution & Logistics Management* 33.6, pp. 531–549. ISSN: 0960-0035. DOI: 10.1108/09600030310492788.

Stadtler, H., Kilger, C., and Meyr, H., eds. (2015). *Supply Chain Management and Advanced Planning*. Springer Texts in Business and Economics. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-55308-0. DOI: 10.1007/978-3-642-55309-7.

Štolba, M. and Komenda, A. (2013). "Fast-forward heuristic for multiagent planning." In: *Proceedings of the 1st Workshop on Distributed and Multi-Agent Planning*. Ed. by R. Nissim, D. L. Kovacs, and R. I. Brafman.

Sycara, K., Paolucci, M., Ankolekar, A., and Srinivasan, N. (2003). "Automated discovery, interaction and composition of semantic web services." In: *Web Semantics: Science, Services and Agents on the World Wide Web* 1.1, pp. 27–46. ISSN: 1570-8268.

The Workflow Management Coalition Specification (1999). *Terminology & Glossary: WFMC-TC-1011 (Issue 3.0)*.

Torreño, A., Onaindia, E., and Sapena, Ó. (2012). "An approach to multi-agent planning with incomplete information." In: *Proceedings of 20th biennial European Conference on Artificial Intelligence (ECAI)* 242.762-767.

Torreño, A., Onaindia, E., and Sapena, Ó. (2014a). "FMAP: Distributed cooperative multi-agent planning." In: *Applied Intelligence* 41.2, pp. 606–626.

Torreño, A., Onaindia, E., and Sapena, Ó. (2014b). "Integrating individual preferences in multi-agent planning." In: *Proceedings of the 2nd Workshop on Distributed and Multi-Agent Planning*. Ed. by D. Borrajo, D. L. Kovacs, and A. Torreño.

Van der Aalst, W. M. P. (1999). "Interorganizational Workflows: An Approach Based on Message Sequence Charts and Petri Nets." In: *Systems Analysis - Modelling - Simulation* 34.3, pp. 335–367.

Van der Aalst, W. M. P. (2015). "Extracting Event Data from Databases to Unleash Process Mining." In: *BPM - Driving Innovation in a Digital World*. Ed. by J. vom Brocke and T. Schmiedel. Management for Professionals. Cham: Springer International Publishing, pp. 105–128. ISBN: 978-3-319-14429-0. DOI: 10.1007/978-3-319-14430-6_8.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Van der Aalst, W. M. P. and van Hee, K. M. (2002). *Workflow Management: Models, methods and systems*. Cambridge, MA: MIT press.

Van der Aalst, W. M. P., Weijters, A. J. M. M., and Maruster, L. (2004). "Workflow mining: Discovering process models from event logs." In: *IEEE Transactions on Knowledge and Data Engineering* 16.9, pp. 1128–1142.

Wang, P., Ding, Z., Jiang, C., and Zhou, M. (2014). "Automated web service composition supporting conditional branch structures." In: *Enterprise Information Systems* 8.1, pp. 121–146. ISSN: 1751-7575. DOI: 10.1080/17517575.2011.584132.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Weber, I., Hoffmann, J., and Mendling, J. (2008b). "Semantic business process validation." In: *Proceedings of the 3rd International Workshop on Semantic Business Process Management*.

Wetzstein, B., Ma, Z., Filipowska, A., Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., and Cicurel, L. (2007). "Semantic Business Process Management: A Lifecycle Based Requirements Analysis." In: *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007) in conjunction with the 3rd European Semantic Web Conference (ESWC 2007)*.

Wohed, P., van der Aalst, W. M. P., Dumas, M., ter Hofstede, A. H. M., and Russell, N. (2006). "On the Suitability of BPMN for Business Process Modelling." In: *Business Process Management*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, S. Dustdar, J. L. Fiadeiro, and A. P. Sheth. Vol. 4102. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 161–176. ISBN: 978-3-540-38901-9. DOI: 10.1007/11841760_12.

Wooldridge, M. J. (2009). *An introduction to multiagent systems*. 2nd ed. Chichester: John Wiley & Sons Ltd.

Ou-Yang, C. and Winarjo, H. (2011). "Petri-net integration – An approach to support multi-agent process mining." In: *Expert Systems with Applications* 38.4, pp. 4039–4051. ISSN: 09574174. DOI: 10.1016/j.eswa.2010.09.066.

Zhang, J. (2017). "The Technical Foundation of a Multi-Agent System." In: *Multi-Agent-Based Production Planning and Control*. John Wiley & Sons Singapore Pte. Ltd, pp. 21–53. ISBN: 9781118890073. DOI: 10.1002/9781118890073.ch2.

3

Automated Adaptation of Existing Process Models

This Chapter addresses **FT2** and RQ2 in particular. In contrast to the previous Chapter, the adaptation of existing process models by means of Automated Planning is envisioned. This is of particular interest as processes nowadays are strongly influenced by IF1 upcoming needs for change. Paper 4, which focusses of the particular question how process models can be adapted to upcoming needs for change in advance by means of an Automated Planning approach.

3.1 Paper 4: Adapting Process Models via an Automated Planning Approach

Status	Submitted	Full Citation
in revision	12/2019	Heinrich, B., Schiller, A., Schön, D. and Szubartowicz, M. (2020), Adapting Process Models via an Automated Planning Approach, Working Paper, University of Regensburg, 2020.

Abstract

Today's fast-paced business world poses many challenges to companies. Amongst them is the necessity to quickly react to needs for change due to shifts in their competitive environment. Hence, a high flexibility of business processes while maintaining their quality has become a crucial success factor. We address this issue by proposing an automated planning approach that is capable of adapting existing process models to upcoming needs for change. This means that the needs for change have not yet been implemented and the adapted process models have so far not yet been realized. Our work identifies and addresses possible changes to existing process models. Further, it provides adapted process models, which are complete and correct. More precisely, the process models resulting from the presented approach contain all feasible and no infeasible paths. To enable an automated adaptation, the approach is based on enhanced methods from automated planning. We evaluate our approach by means of mathematical proofs and an application in a real-world situation. Further, by means of a simulation experiment, its runtime is benchmarked against planning process models from scratch.

Keywords: Process Flexibility, Process Changes, Process Models, Business Process Management

Post submission changes:

- Several formatting changes for consistency reasons
- Changed citation style for consistency reasons
- The numbering of headlines, tables, figures and definitions was changed for consistency reasons
- Renamed contributions from (C1) and (C2) to ❶ and ❷ for consistency reasons
- Consistent capitalization of Section references

3.1.1 Introduction

The ability to be agile and align existing capabilities to new needs quickly is one of the most important factors for companies' success and competitive advantage (McElheran, 2015). Companies are required to react to shifts in their competitive environment flexibly and within short time in order to stay operational and competitive (Döhring et al., 2014; Forstner et al., 2014; Katzmarzik et al., 2012; Reisert et al., 2018; Rosemann et al., 2015). Examples of such shifts include dynamic customer behavior, market developments or new regulatory requirements and are referred to as needs for change. According to Le Clair (2013), in the last decade the inability to react to such needs for change has led to 70% of the Fortune 1000 companies to be removed from this list. The study proposes ten dimensions to characterize business agility, half of them being process-focused. This underlines that improving the flexibility of business processes while maintaining their quality has become a crucial success factor for companies (Reichert et al., 2012). Here, process flexibility is understood as the ability to configure or adapt a process and the according process model without completely replacing it (Afflerbach et al., 2014; Bider, 2005; Hallerbach et al., 2010; Regev et al., 2007). It is hardly surprising that the importance of process flexibility is widely recognized in the literature (cf., e.g., Cognini et al., 2018; Ellis et al., 1995; Hammer, 2015; Hull et al., 2016; La Rosa et al., 2017; Mejri et al., 2018; van der Aalst, 2013).

Process flexibility is also acknowledged as an important issue in practice. To give an example, in an extensive project with a European bank we analyzed over 600 core business processes as well as 1,500 support processes. These processes spread across different departments and business areas of the bank. The majority of the processes and their corresponding process models, which initially had been modeled using the ARIS toolset, required a frequent redesign or adaptation due to needs for change caused by, for instance, new or enhanced distribution channels and changing products. Indeed, the bank has been conducting projects to adapt business process models much more frequently, causing the vast majority of the budget, compared to projects to design completely new business process models. Moreover, several IT and business executives of the bank stated that nowadays process redesign projects are more time-consuming than they were ten years ago due to a higher complexity. Interviews with staff members of companies in other industries supported these insights. This underlines the relevance of approaches for an adaptation of process models.

The increasing complexity of process models and process (re)designs also has another effect: Constructing and adapting process models manually turns out to be a more and more difficult task. According to Mendling et al. (2008), especially larger and more complex process models are likely to contain more errors when constructed manually.

For instance, Roy et al. (2014) and Fahland et al. (2011) examined business process models in an industrial context and found that up to 92.9% of these models contained at least one (syntactical or semantic) error. Hence, in this paper, we follow other approaches making use of automation techniques (e.g., algorithms) when modeling processes (e.g., Marrella, 2019; Rosemann et al., 2010). The research strand "automated planning of process models" (Heinrich et al., 2015b; Heinrich et al., 2015c; Henneberger et al., 2008; Hoffmann et al., 2009; Hoffmann et al., 2012) aims to construct process models in an automated manner and from scratch. Here, a process model is planned by means of algorithms based on states, actions, and control flow patterns. Our approach for an automated adaptation of process models enhances methods of automated planning and thus contributes to this research strand.

Adapting process models to needs for change comprises the modeling of an existing or desired process. Modeling an existing process means that the required changes have already been realized. In this case, the changed process can be modeled by adapting or reconstructing an existing process model to new records of event logs not already considered in the existing process model (cf. process enhancement; IEEE Task Force on Process Mining, 2012; Kalenkova et al., 2017). In contrast, modeling a desired process means that the needs for change have not yet been implemented and the desired process models have so far not yet been realized. In this paper, we focus on the latter perspective, thus aiming to adapt existing process models to needs for change *in advance* and to construct models of desired processes, leading to the following research question:

> How can process models be adapted to needs for change in advance in an automated manner?

In literature, many existing approaches for the adaptation of process models aim to "repair" process models locally when considering changes (e.g., Alférez et al., 2014; Eisenbarth, 2013). However, both process models and process (re)designs are becoming increasingly large and complex (cf. Hornung et al., 2007 and the discussion above) and local repairs or changes to just some components of process models are not sufficient. Instead, the challenging task of providing adapted process models, which are *correct* and *complete*, has to be addressed. Correct means that the adapted process models contain *only* feasible paths and no infeasible paths, while complete means that the adapted process models contain *all* feasible paths. Correct and complete process models are important to, for instance, increase "flexibility by definition", which is "the ability to incorporate alternative execution paths within a process definition at design time such that selection of the most appropriate execution path can be made at runtime for each process instance" (van der Aalst, 2013). A correct and complete process model enables

the flexibility to select the most appropriate feasible path for execution (e.g., based on economic criteria). Hence, we discuss the following research question:

> How can process models be adapted such that the resulting process models are correct and complete?

The main contributions of this paper are thus as follows:

❶ *Adaptation to needs for change in advance in an automated manner.* The approach adapts existing process models to needs for change in advance (i.e., no reconstruction of existing process models, e.g., to new records of event logs). To this end, it enhances methods especially from automated planning of process models.

❷ *Construction of correct and complete process models.* The approach adapts process models in such a way that the resulting process models are correct and complete.

In the next Section, we discuss related work to explicate our research gap. Thereafter, we introduce a running example and define the formal foundation, which forms the basis of our approach. After that, we present our approach to adapt existing process models to needs for change in advance via automated planning. Subsequently, we evaluate our approach by means of mathematical proofs of its key properties, demonstrate its efficacy by means of an application in a real-world situation and benchmark its performance in a simulation experiment. We conclude by summarizing our work, discussing its limitations and proposing future research.

3.1.2 Related Work

In the following, we will discuss existing approaches dealing with an adaptation of process models. To structure this discussion, we consider five phases of the BPM lifecycle as proposed by vom vom Brocke et al. (2018) and omit the process identification phase, as it is not subject of our research. We start with approaches in (1) the process discovery phase and continue by discussing existing approaches in (2) the process analysis phase. Thereafter, we briefly analyze approaches in (3) the process re-design phase, (4) the process implementation phase and close with (5) the process monitoring and controlling phase. Table 3.1 at the end of the Section summarizes our discussion.

Ad (1): During the process discovery phase, detailed information about processes (e.g., in terms of process models) is derived from actually conducted processes in a company. The research field of process mining addresses the area of process discovery (cf., e.g., Augusto et al., 2019; IEEE Task Force on Process Mining, 2012; van der Aalst, 2015). Within this area, approaches use event logs from process instances to reconstruct process models (van Dongen et al., 2009). The issue that the information of

event logs might change from time to time due to changes in the corresponding executed process, which needs to be considered when discovering the process model, has extensively been addressed in the literature (Bose et al., 2014; Chen et al., 2012; Meihong et al., 2012). The focus of this research, however, is different to ours because an *existing*, already changed process is *re*constructed. This means, the aim is a reconstruction of a (new) process model considering needs for change already realized in actual process instances. Therefore, these works do not aim to provide an approach for adapting process models to changes in advance and, as they rely on event logs, do not present concepts to support this task. In contrast, we aim to model a desired process which is not yet realized and thus to adapt to needs for change in advance (cf. ❶).

Ad (2): During the process analysis phase, for instance, weaknesses in the discovered processes are determined. In this context, the research field of process (model) and workflow verification (e.g, Masellis et al., 2017) aims to check and improve syntactic and semantic correctness of process models. For instance, the automated repair of unsound workflow nets by means of annealing procedures (i.e., heuristic approaches which generate a set of alternative workflow nets containing fewer errors) is envisioned by Gambini et al. (2011). Further, the verification of workflows by means of Petri nets is focused on by Verbeek et al. (2005) and Wynn et al. (2009) in order to "detect the soundness property". However, within this research field, there is no work on the adaptation of process models to needs for change in advance (cf. ❶).

Ad (3): Approaches in the process re-design phase aim to increase process flexibility in the way they model business processes, for instance by capturing customizable process models (La Rosa et al., 2017). To this end, manual as well as automated (i.e., by means of an algorithm) approaches have been proposed. Generalizations (van der Aalst et al., 2009; vom Brocke, 2009) or specific change patterns (Weber et al., 2008a), which are both constructed manually, provide possibilities to increase process flexibility. Generalization approaches result in less specific process models, due to, for instance, the assignment of several specific actions to one abstract, general action. Hence, such generalized process models (e.g., reference models; cf. vom Brocke, 2009) may lack support for real-world scenarios that are not modeled explicitly, especially with respect to an (automated) process execution (cf., e.g. Khan et al., 2010; Weber, 2007). On the other hand, specific change patterns allow the replacement of parts of a process model – often supported by a modeling tool – by different, predesigned parts. The purpose of those approaches is different from ours, since they do not aim to provide an approach for the automated adaptation of process models, which are correct and complete (cf. ❶ and ❷). A second research strand in the process re-design phase striving to increase process flexibility is the automated planning of process models (cf., e.g., Heinrich et al., 2018b). In this strand, few approaches exist that address the issue of adapting process models to

needs for change in advance (cf. Eisenbarth, 2013; Eisenbarth et al., 2011; Lautenbacher et al., 2009). These works adapt parts of a process model due to (a few) changed actions. They identify so-called single-entry-single-exit fragments surrounding an action to be changed. Based on such an identified fragment, the idea is to determine "quasi-" initial and goal states (for the considered fragment) and to initiate a regular process planning in order to replace the existing fragment by means of a newly planned fragment. Changed actions, however, can affect the whole process model (e.g., when a changed action results in several new feasible paths), so that the process models adapted by these approaches are usually not complete. Further, these approaches do not ensure that the whole process model is correct because of adapting only fragments. Additionally, the need for adapting a process model may not only arise from actions to be changed but also from changed initial and goal states, which is not covered by this research. To sum up, these works do not aim to provide adapted process models which are complete and correct and are "interested in adapting only parts of a model" (Eisenbarth et al., 2011), in contrast to ❷.

Ad (4): During the process implementation phase, the previously (in the process redesign phase) constructed process model is implemented in the according execution systems. For instance, (web) services are composed with the aim of aggregating existing functionality into new functionality. For this, graph structures consisting of services and states, which are similar to actions and states in process models, are constructed. Thus, within the research field of (web) service composition, issues similar to the adaptation of process models are discussed as "network configurations and QoS [Quality of Service] offerings may change, new service providers and business relationships may emerge and existing ones may be modified or terminated" (Chafle et al., 2006). Here, research focuses on replacing (web) services (or small combinations of services) by other, functionally equivalent (small combinations of) services (cf., e.g., Bucchiarone et al., 2011; Canfora et al., 2005). Within this research field, some authors use so-called variability models, which are very similar to the change patterns mentioned above, to adapt service compositions (Alférez et al., 2014; La Rosa et al., 2017). However, in contrast to these approaches, our considered changes regarding process models are not limited to exchanging (a few) actions but rather we aim to adapt whole process models in such a way that the resulting process models are correct and complete (cf. ❷).

Ad (5): In the process monitoring and controlling phase, several works exist that envision to use so-called continuous planning for the recovery of failed process executions (cf., e.g., Linden et al., 2014; Marrella et al., 2011a; Marrella et al., 2011b; Marrella et al., 2012; Tax et al., 2017; van Beest et al., 2014). These works aim for error handling procedures that are based on planning techniques in order to resolve process executions interrupted due to, for instance, external events. Other works support users by provid-

ing change operations to address ad-hoc deviations from pre-modeled task sequences within a workflow (Reichert et al., 1997; Reichert et al., 1998; Rinderle et al., 2004). However, they do not propose an approach to enable the adaptation of process models (cf. ❶). In particular, as these works aim to address particular process instances, they do not strive to provide complete process models for the business process as a whole (cf. ❷). Another kind of approaches (cf., e.g., Garrido et al., 2010; Gerevini et al., 2000; Gerevini et al., 2012; Kambhampati, 1997; Marrella et al., 2017; Nunes et al., 2018; Scala et al., 2015; van der Krogt et al., 2002; van der Krogt et al., 2005) that make use of planning algorithms deals with the issue of adapting a process model due to discrepancies which occurred during the conduction. Here, the task is to find a sequence of actions that will resolve the misalignment between the modeled and the actual reality Marrella et al. (2017). Similarly, Kambhampati (1997) introduces the concept of refinement planning as "the process of starting with the set of all action sequences and gradually narrowing it down to reach the set of all solutions". Here, so-called candidates (i.e., parts of a plan consistent with certain constraints) are combined to subsequently construct a feasible complete plan. Further, Gerevini et al. (2000), for instance, propose a fast plan adaptation by identifying delimited parts of the plan that are inconsistent and then replanning the subgraph for these delimited parts. In the worst case, these parts comprise the whole plan, making planning from scratch necessary. However, they do not aim to adapt whole process models (cf. ❷). Further, these approaches tend to address *momentary* changes that "occur on an individual or selective basis" (van der Aalst et al., 2000b). However, we aim to address both momentary and *evolutionary* (*permanent*) changes that "are of a structural nature" and are typically "forced by legislature or changing market demands" (e.g., van der Aalst et al., 2000b). Nebel et al. (1995) provide an empirical analysis about the efficiency of plan reuse versus (new) plan generation. They compare the worst-case complexity of planning from scratch with reusing and modifying plans (so-called planning from second principles). The authors state that planning from second principles consists of two steps: The identification of an appropriate plan candidate from a plan library and its modification so that it solves a new problem instance. As they aim for a "minimal modification of a plan", they do not strive to construct complete process models (cf. ❷). In this regard, there exist a few declarative process modeling approaches that address similar issues as well. Declarative process models are an alternative to the (imperative) process models addressed in this paper, specifying what should be done in a process, not how (Pesic et al., 2007; Pesic et al., 2006; van der Aalst et al., 2009). They tend to address *momentary* changes, whereas we aim to address both momentary and *evolutionary* changes (van der Aalst et al., 2000b). For declarative process models, it is further proposed to generate so-called "optimized enactment plans" that could be understood as a planning problem (cf., e.g., Barba et al., 2013a; Jiménez-Ramírez et al.,

2013). In this context, a replanning approach is envisioned by Barba et al. (2013b), in case the actually conducted process deviates from the generated optimized enactment plan. However, they aim at "optimizing performance goals like minimizing the overall completion time" in contrast to ❷ and do not adapt to needs for change in advance (cf. ❶).

Finally, the research field of process mining comprises the areas of conformance checking and process enhancement (IEEE Task Force on Process Mining, 2012; Leemans et al., 2018; van der Aalst, 2015) that are also part of the process monitoring and controlling phase. Conformance checking is used to detect differences between the traces of a process execution (e.g., found in event logs) and a given process model (de Leoni et al., 2017; García-Bañuelos et al., 2018; van der Aalst et al., 2014). In process enhancement (which deals with tasks such as "model extension" or "model repair"), the goal is to change or extend an already existing process model by taking information about the process instances from event logs into account (cf., e.g., Fahland et al., 2012). The focus of this research, however, is different to ours because an *existing*, already instantiated and enacted process is analyzed with respect to deviations from an *existing* process model. This means, the aim is an adaptation of a process model to needs for change already realized in actual process instances. Therefore, these works do not aim to provide an approach for adapting process models to changes in advance as they rely on event logs. In contrast, we aim to model a desired process which is not yet realized and thus to adapt to needs for change in advance (cf. ❶).

To sum up, to the best of our knowledge, there is no existing approach that adapts process models to needs for change in advance in an automated manner (cf. ❶) and constructs correct and complete process models (cf. ❷).

3.1.3 Running Example & Formal Foundation

In our research, we aim for a representation of process models independent of a particular process modeling language. More precisely, in contrast to relying on one single process modeling language such as Event-driven Process Chains (EPC), we use a formal foundation that provides a broader application scope for our approach. Our formal foundation includes so-called process graphs, which are also referred to as planning graphs in the research field of automated planning of process models (e.g., Heinrich et al., 2015b; Henneberger et al., 2008; Lin et al., 2012; Zheng et al., 2008). Process graphs utilize similar concepts as existing well-known process modeling languages such as EPCs, Business Process Model and Notation (BPMN) or Unified Modeling Language (UML) activity diagrams (e.g., van Gorp et al., 2013).

To illustrate the formal foundation and our approach, we use a simplified excerpt con-

Lifecycle phase		Time of consideration	Adaptation to upcoming needs for change in advance in an automated manner ❶				Construction of correct and complete process models ❷
			Adaptation to upcoming needs for change in advance		Automated approach		
1) Process discovery		design time	✖	not considered; aiming to reconstruct process models that represent already realized needs for change in processes	✔	considered	✔ considered
2) Process analysis		design time	✖	not considered	✔	considered	○ considered in some selected works
3) Process re-design	Process flexibility	design time	✖	not considered	✖	manual approaches	○ not explicitly considered; it may be expected that correctness is considered implicitly
	Automated planning of process models	design time	○	consider upcoming needs for change but only for single actions/fragments	✔	considered	✖ not aiming to provide a complete and correct process model
4) Process implementation		design and execution time	✖	not considered	○	considered by some selected works	✖ not considered
5) Process monitoring and controlling	Workflow management	execution time	✖	not considered; aiming at error handling during the execution of processes	✔	considered	✖ not considered; not aiming to provide a complete process model
	Planning	execution time	✖	not considered; aiming to address discrepancies that occurred during process execution	✔	considered	✖ not aiming to provide a complete process model
	Declarative process modeling	design time	✖	not considered	○	considered by few works	✖ not aiming to provide a complete process model
	Conformance checking / Process enhancement	design time	✖	not considered; aiming to adapt process models that represent already realized needs for change	✔	considered	✔ considered

Table 3.1: Overview of Related Work

sisting of three actions from a real-world manufacturing process of a European electrical engineering company as a running example (the whole process is part of our evaluation in Section 3.1.5). The process is repeatedly influenced by changing requirements and new legal regulations and therefore needs to be adapted frequently. In a first step, the required material needs to be ordered (action "Order material") as there is no material in stock. Thereafter, a circuit board is prefabricated (action "Prefabricate circuit board"). Here, basically, the circuit board goes through the actions of developing, etching and stripping. In order to produce a complete product, the prefabricated circuit board subsequently needs to be assembled with other parts such as microchips and resistors (action "Assemble product"). Finally, the product is ready for sale and the process terminates. Figure 3.1 shows the process graph of our running example denoted in terms of the formal foundation presented in the following.

The process starts at an initial belief state (short: initial state). *Belief states* are denoted by tables (e.g., in Figure 3.1, the first belief state at the top, annotated with "Initial state"). They comprise multiple pieces of information, so-called *belief state tuples* which are represented by the rows in the according tables. For instance, within our running example, the belief state tuple (*product, not manufactured*) in the upmost table of the process graph in Figure 3.1 (annotated with "Initial state") expresses that at the beginning of the process, the product is not yet manufactured. *Actions* which lead from one belief state to another are denoted by rounded rectangles (e.g., the action "Order material"). Actions contain *preconditions* (denoted by $pre(a)$) and *effects* (denoted by $\mathit{eff}(a)$). Preconditions (including inputs) denote everything an action needs to be applied, whereas *effects* (including outputs) denote everything an action provides, deallocates or alters after it was applied. The process ends at one to possibly many defined belief states meeting a goal state (i.e., the goal of the process is achieved). For example, in the belief state at the very bottom in Figure 3.1, a belief state meeting a goal state is reached because the product is manufactured which represents the defined goal state (*product, manufactured*), denoted in italics.

The essential notions are presented formally in the following Definitions 3.1.1 to 3.1.5. Hereby, we follow common ways to represent a planning domain (Ghallab et al., 2004; Ghallab et al., 2016) within automated planning (Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2015b; Heinrich et al., 2015c; Heinrich et al., 2016; Henneberger et al., 2008). We thus ensure compatibility with existing works.

Definition 3.1.1. (*belief state tuple*). A *belief state tuple* p is a tuple consisting of a *belief state variable* $v(p)$ and a subset $r(p)$ of its *domain* $dom(p)$, which we will write as $p{:=}(v(p),r(p))$. The domain $dom(p)$ specifies which values can generally be assigned to $v(p)$ and can for instance represent a data type such as *integer* or a finite set. The set

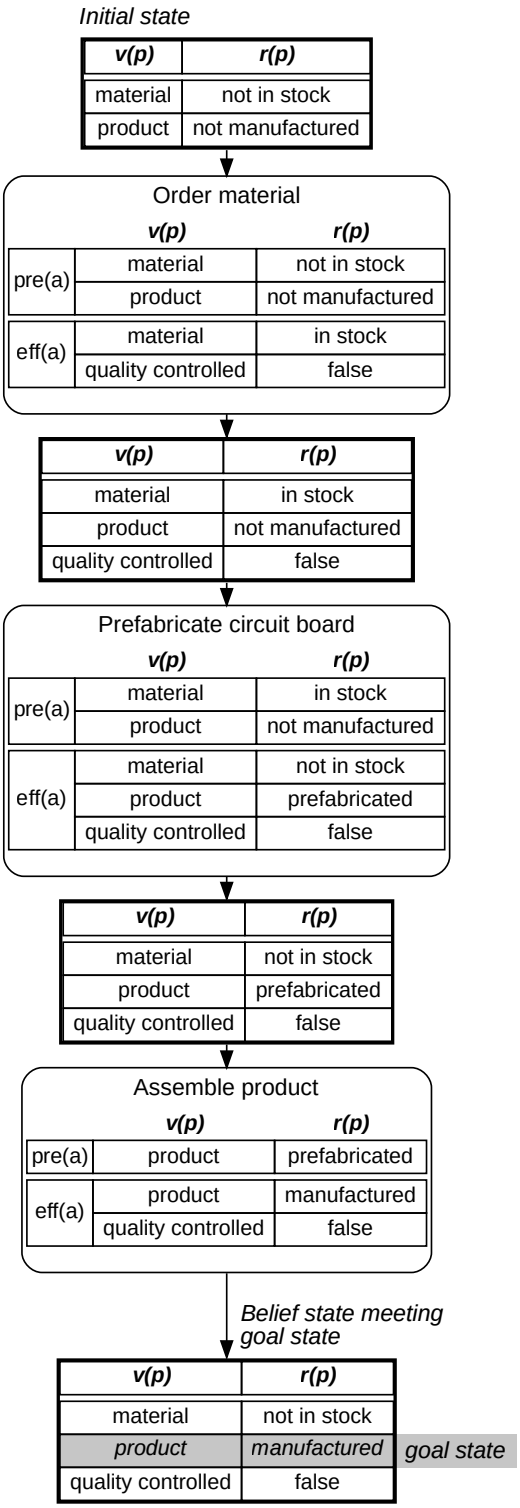

Figure 3.1: Process Graph of the Simplified Manufacturing Process

$r(p) \subseteq dom(p)$ is called the *restriction* of $v(p)$ and contains the values that can be assigned to $v(p)$ in this specific belief state tuple p. Let $BST=\{p_1,\ldots,p_n\}$ be a finite set of belief state tuples.

Definition 3.1.2. (*action*). An *action* a is a triple consisting of the action name and two sets, which we write as $a:=(name(a),pre(a),eff(a))$. The set $pre(a) \subseteq BST$ are the *preconditions* of a and the set $eff(a) \subseteq BST$ are the *effects* of a. An action a is *applicable* in a belief state bs iff $\forall w \in pre(a)\ \exists u \in bs$: $v(w)=v(u) \wedge r(w) \cap r(u) \neq \varnothing$. In other words, a is applicable in bs iff all belief state variables in $pre(a)$ also exist in bs and the respective restrictions of the belief state variables intersect.

Belief state tuples and actions are used in the definition of a nondeterministic belief state-transition system presented in the following. The graph in Figure 3.1 is based on such an underlying nondeterministic belief-state transition system. Here, the initial state contains the two belief state tuples (*material*, {*not in stock*}) and (*product*, {*not manufactured*}) with *material* and *product* being the belief state variables and {*not in stock*} and {*not manufactured*} being their restrictions. A nondeterministic belief state-transition system is defined in terms of its belief states, its actions and a transition function that describes how an action leads from one belief state to possibly many belief states (Bertoli et al., 2006; Ghallab et al., 2004; Ghallab et al., 2016).

Definition 3.1.3. (*nondeterministic belief state-transition system*). A *nondeterministic belief state-transition system* is a tuple $\sum=(BS,A,R)$, where

i. $BS \subseteq 2^{BST}$ is a finite set of *belief states*. A *belief state* $bs \in BS$ is a subset of BST, containing every belief state variable one time at the most.

ii. A is a finite set of actions. The set of actions that are applicable in bs are denoted by $app(bs):=\{a \in A \mid a \text{ is applicable in } bs\}$.

iii. $R: BS \times A \rightarrow 2^{BS}$ is the transition function. For each belief state $bs \in$ BS and each action $a \in A$ applicable in bs the set of next belief states is calculated as $R(bs,a)=bst_{old} \cup bst_{pre(a)} \cup eff(a)$. Here, $bst_{old}=bs \setminus \{(v(t),r(t)) \in bs \mid \exists (v(s),r(s)) \in pre(a) \cup eff(a)$: $v(t)=v(s)\}$ are the belief state tuples of bs that are determined by the transition function to remain unchanged (the notation "\" represents the set-theoretic difference).

   Furthermore, $bst_{pre(a)}=\{(v(t), r(t) \cap r(s)) \mid (v(t),r(t)) \in bs \wedge (\exists (v(s),r(s)) \in pre(a)$: $v(t)=v(s)) \wedge (\nexists (v(x),r(x)) \in eff(a)$: $v(t)=v(x))\}$ are the belief state tuples of bs whose restriction is further limited by the preconditions of a. If a is not applicable in bs, $R(bs,a)=\varnothing$.

Based on this definition, a graph as presented in Figure 3.1 and defined in Definition 3.1.5 can be constructed from scratch by means of existing planning approaches (cf., e.g., Bertoli et al., 2006; Ghallab et al., 2004; Ghallab et al., 2016; Heinrich et al., 2009; Heinrich et al., 2015c). The planning starts with an initial state, constructs the following belief state for each applicable action based on the transition function $R(bs,a)$ and continues until a goal state is met (e.g., in Figure 3.1, the goal state (*product, manufactured*) written in italics is met by the belief state at the very bottom). The input data for the planning can, for instance, be obtained by extracting actions from existing process models, using interfaces of process modeling tools, fresh modeling of actions or conceptualization of (web) services (Bortlik et al., 2018; Heinrich et al., 2018a).

Definition 3.1.4. (*goal state*). A *goal state* is a subset of *BST*, containing every belief state variable one time at the most, which represents a termination criterion for the process. If a belief state *bs* fulfills the termination criterion represented by a goal state *goal* (i.e., $\forall p \in goal: \exists p' \in bs, v(p)=v(p'), r(p') \subset r(p)$), we denote *bs* as *meeting goal*.

Definition 3.1.5. (*process graph*). A *process graph* is a bipartite, directed, finite graph $G=(N,E)$ with the set of nodes N and the set of edges E. The set of nodes N consists of two partitions: The set of action nodes A and the set of belief state nodes BS. Each node $bs \in BS$ represents one distinct belief state in the process graph. Each action node $a \in A$ represents an action in the process graph. The process graph starts with one initial state $bs_{\text{init}} \in BS$ and ends with one to possibly many belief states $bs_{\text{goal,j}} \in BS$ meeting a goal state. A (finite) sequence of states and actions $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ starting with the initial state is called a *path*. A path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ is called a *feasible path* if the following three additional conditions apply:

i. bs_{k} meets a goal state

ii. $bs_{\text{init}},\ldots,bs_{\text{k-1}}$ do not meet any goal state

iii. $a_1 \in app(bs_{\text{init}})$, $bs_2=R(bs_{\text{init}},a_1)$,..., $a_{\text{k-1}} \in app(bs_{\text{k-1}})$, $bs_{\text{k}}=R(bs_{\text{k-1}},a_{\text{k-1}})$.

Within this paper, we present an approach to adapt process graphs as described in Definition 3.1.5. The result of this adaptation is again a process graph based on the definitions presented above. Hence, existing works for the automated construction of control flow patterns (van der Aalst et al., 2003) such as exclusive choice based on process graphs (e.g., Heinrich et al., 2015b; Heinrich et al., 2016; Meyer et al., 2006) can be used as usual to construct process models containing control flow patterns. Thus, it is not necessary to address how to consider control flow patterns in this paper.

3.1.4 Design of our Approach

We propose two steps for the adaptation of process graphs. In the first step (i), we identify and classify possible changes to a given process graph. In the second step (ii), potential consequences resulting from these identified changes are addressed.

Ad (i): As given in Definition 3.1.5, a process graph consists of belief states (amongst them one initial state and one to possibly many belief states meeting a goal state) and actions. When constructing a process graph using an existing approach for the automated planning of process models, the initial state, the goal states and the actions are used as input. All other belief states of the process graph are constructed during planning and thus, it is not possible without creating inconsistencies that these belief states are *directly* adapted due to needs for change. Therefore, by using our formal foundation, every need for change to a process graph is reflected in a change to this input and consequently, only changes to this input need to be considered for the adaptation of process graphs.

The initial state, goal states and preconditions and effects of actions are all represented by sets of belief state tuples. When changing these sets we will consider changes to single belief state tuples in the following as changes to multiple belief state tuples can be represented by a sequence of changes to single belief state tuples. In this way, we aim to establish so-called *atomic changes*. These are changes that can represent every adaptation to a process graph when put into sequence.

To identify atomic changes, we align our research to the well-known CRUD operations. The CRUD operations have their origin in database systems (cf. Martin, 1983) and are the four elemental, low-level operations "create", "read", "update" and "delete" that cover all possible ways of accessing and altering data. We determine all atomic changes presented in Table 3.2 by combining these operations with the input discussed above. As "read" does not represent a change in our context, this operation is not taken into account in Table 3.2.

Since exactly one initial state is used as input for planning, each change to the initial state can be represented by an update of it. For goal states and actions as well as for each belief state tuple, however, the operations "create", "update" and "delete" can be applied.

We note that updating a goal state or action could be treated as deleting the old, existing goal state or action and adding a new (the updated) one. However, reusing existing information about the process graph by considering the operation "update" enables a more efficient approach (cf. Section 3.1.5). For instance, when updating an action and retrieving the belief states in which the updated action is applicable, it may be benefi-

		CRUD operations		
		Create	Update	Delete
Input for planning	Initial State	—	• Add new belief state tuple • Alter existing belief state tuple • Remove existing belief state tuple	—
	Goal States	Add new goal state	• Add new belief state tuple • Alter existing belief state tuple • Remove existing belief state tuple	Remove existing goal state
	Actions	Add new action	• Update preconditions – Add new belief state tuple – Alter existing belief state tuple – Remove existing belief state tuple • Update effects – Add new belief state tuple – Alter existing belief state tuple – Remove existing belief state tuple	Remove existing action

Table 3.2: Overview of Atomic Changes

cial to take into account in which belief states the action was applicable before it was updated.

To address any adaptation of a process graph, a sequence of the presented atomic changes can be used. To give an example, adding multiple single actions sequentially allows to address changes which include a larger number of added actions. A more detailed example is discussed in Section 3.1.4.4.

Ad (ii): We identify potential consequences for the process graph (e.g., actions becoming applicable in an updated initial state of the graph) resulting from each of the discussed atomic changes in the following Sections. Thereby, we do not merely reduce each atomic change to a planning problem solvable by existing techniques for the automated planning of process models (e.g., Heinrich et al., 2015b; Heinrich et al., 2015c; Henneberger et al., 2008; Hoffmann et al., 2009; Hoffmann et al., 2012; Lin et al., 2012; Zheng et al., 2008). Instead, we enhance these techniques to address each individual atomic change compared to "planning from scratch". To do this, we determine where parts of the existing process graph can be reused or where new belief states and actions have to be planned. In addition, we incorporate knowledge about the applicability of actions in the existing process graph to reduce the effort of verifying the applicability of these actions to changed belief states. Please note that a pseudo code of our approach is available in Appendix 7.4.2.

3.1.4.1 Updating the Initial State

Following Definition 3.1.5, a process graph starts with exactly one initial state. Thus, every possible change regarding the initial state, seen as an ordered set of atomic changes, consists of the addition of a belief state tuple to the initial state, the removal of a belief state tuple that was present in the initial state, or the update of a belief state tuple's restriction (cf. Table 3.2). A belief state tuple p with empty restriction in a belief state (i.e., $r(p) = \varnothing$, no value of the belief state variable is feasible) is equivalent to a non-existing belief state tuple. Therefore, the addition of a belief state tuple p can be seen as an update of $(v(p),r(p))$ in which $r(p) = \varnothing$ is changed so that $r(p) \neq \varnothing$ and the removal of a belief state tuple p can be seen as an update of $(v(p),r(p))$ in which $r(p) \neq \varnothing$ is changed to $r(p) = \varnothing$. Thus, we subsequently only need to consider the single case of an updated belief state tuple to fully cover the three possible atomic changes regarding the initial state.

To be able to clearly address the initial state before and after the adaptation, we denote the initial state in the given (i.e., not adapted) process graph with bs_{init} and the initial state after the adaptation with bs_{init}'. As we outline the approach of adapting a process

graph to an updated initial state in detail, it is necessary to distinguish between old, (completely) new and updated states in the process graph:

Definition 3.1.6. (*old, new, updated states*). Let BS be the set of belief states in the given process graph and BS' be the set of belief states in the adapted process graph. Each belief state $bs \in BS$ is called an *old state*.

We denote $bs' \in BS'$ as the *update* of $bs \in BS$ (or generally as *updated*), if all of the following criteria are fulfilled:

i. $bs' \notin BS$ (i.e., bs' is not old)

ii. there is a sequence of actions $a_1, a_2, \ldots, a_k$ in the given process graph so that a_1 is applicable in the initial state bs_{init} ($a_1 \in app(bs_{init})$, the set of actions applicable in bs_{init}, cf. Definition 3.1.2), $bs_1 = R(bs_{init}, a_1)$, $a_2 \in app(bs_1)$ and so forth until $bs = R(bs_{k-1}, a_k)$

iii. this same sequence of actions remains applicable in the adapted process graph (considering the updated initial state) and applying this sequence yields bs'

We call belief states $bs \in BS'$ that are neither old nor updated states *new states*. In other words, if the belief state $bs \in BS'$ is an old state, it is a belief state that was already contained in the given process graph, without any change. If bs is an updated state, it is a belief state that was not contained in the given process graph, but is yielded by a sequence of actions already contained in the given process graph. A new state is a state that was not contained in the given process graph and that is yielded by sequences of actions not contained in the given process graph.

Now we will identify potential consequences resulting from updating the initial state bs_{init}. Updating a belief state tuple p of bs_{init} can impact whether an action a is applicable in bs_{init}' if a belief state tuple p' is contained in the preconditions of a such that $v(p') = v(p)$ (cf. Definition 3.1.2). Otherwise, the belief state tuple p is not relevant in order to determine whether a is applicable. Hence, the sets $app(bs_{init})$ and $app(bs_{init}')$ can only differ in actions containing a belief state tuple p' with $v(p) = v(p')$ in their preconditions. Thus, for the set of actions $\{a \in app(bs_{init}) | \nexists p' \in pre(a)\colon v(p') = v(p)\}$, the applicability regarding bs_{init}' does not need to be checked as these actions are unaffected and thus remain applicable. Actions in the set $\{a \in A | \exists p' \in pre(a)\colon v(p') = v(p)\}$, however, need to be checked for potential applicability in bs_{init}'. The actions not contained in $app(bs_{init}')$ are not planned at this point in the adapted process graph.

For each action a that is applicable in both bs_{init} and bs_{init}' (i.e., $a \in app(bs_{init}') \cap app(bs_{init})$) and hence "retained" its applicability we can use $bs = R(bs_{init}, a)$ from the given graph, which helps us to determine $bs' = R(bs_{init}', a)$ as we only need to apply the transition

function R (cf. Definition 3.1.3) with respect to p and transfer these effects to bs. If bs' was contained in the given process graph and thus is an old state, we can retain the whole subgraph starting with bs' as the actions that can be applied in this belief state are known from the given process graph and do not differ since both the belief state and the actions did not change. This is in accordance to existing techniques for the automated planning of process models where the traversal of a previously known state terminates planning. Otherwise (i.e., if bs' is not an old state) bs' is the update of bs (cf. Definition 3.1.6). In this case, the updated belief state can now either meet a goal state, which completes the path, or we need to continue by treating bs' as we currently handle $bs_{\text{init}}{}'$.

The set $app(bs_{\text{init}}{}')$, however, can also contain actions that were not applicable in bs_{init}. For each such action $a \in app(bs_{\text{init}}{}')$ with $a \notin app(bs_{\text{init}})$, the transition function R needs to be applied entirely (i.e., not only with respect to the updated belief state tuple p) in order to obtain the belief state $bs = R(bs_{\text{init}}{}', a)$. If bs meets a goal state, the path is completed, else bs is either old, updated (from a hitherto feasible path) or new. In the first case, we retain the whole subgraph starting with bs from the given process graph. If, however, bs is a new state, we have to apply the transition function R entirely: We compute $app(bs)$ and, for each $a \in app(bs)$, the belief state $R(bs, a)$ following bs. Again, these belief states have to be checked in regard to being old, updated or new. Updated states are handled in the same way as $bs_{\text{init}}{}'$. We proceed iteratively in this manner with every upcoming state depending on its classification regarding Definition 3.1.6.

Altogether, the approach – in line with existing approaches for the automated planning of process models – starts with the initial state, aborting the traversal of a path as soon as an old state, a belief state which meets a goal state or a belief state bs with $app(bs) = \varnothing$ is reached. Especially when traversing updated states it poses an improvement to existing techniques for the automated planning of process models as information from the initial process graph is (re)used.

Within the example (cf. Figure 3.2; parts influenced by the adaptation are black, not influenced parts are grey), a new external supplier that meets the service level requirements is acquired as a business partner. This external supplier is able to provide prefabricated circuit boards. Hence, the fact that now an appropriate external supplier is available is denoted in terms of the belief state tuple (*external supplier,* {*available*}), which therefore is added in the initial state (bold). By means of this change, the action "Order prefabricated circuit board" (retrieved from the set of actions A, cf. Definition 3.1.3), which requires this particular belief state tuple, becomes applicable and thus is planned in the adapted initial state. After this action, a new belief state is created in which the action "Assemble product" is applicable, which in turn leads to the goal state. Thus,

as result of the adaptation, a new feasible path (denoted by means of bold arrows and bold-bordered actions and belief states) is constructed.

3.1.4.2 Changing (the Set of) Goal States

A process graph contains one to possibly many goal states (cf. Definition 3.1.5). In alignment with the CRUD functions, we consider the atomic changes "adding a goal state", "removing a goal state" and "updating a goal state" (cf. Table 3.2).

3.1.4.2.1 Adding a Goal State. Denoting the set of all goal states in the given process graph with *GOALS*, the addition of a new goal state $goal \notin GOALS$ with $GOALS' = GOALS \cup \{goal\}$ could, on the one hand, result in new feasible paths, which lead to this new goal state. Such new feasible paths have not been feasible in the given process graph and thus need to be newly constructed. On the other hand, as goal states serve as termination criteria, this new goal state could imply feasible paths in the given process graph being "shortened" so that for a given path $bs_{\text{init}}, a_1, bs_1, a_2, \ldots, bs_k$ there exists $j<k$ with bs_j meeting *goal*.

To determine these consequences, we traverse the paths of the given process graph and their belief states (except for the belief states meeting a goal state from *GOALS* at the end of each such feasible path), starting with the initial state. For each belief state *bs*, we need to check whether *bs* meets the new goal state (first case) or whether actions applicable in *bs* lead to the new goal state subsequently (second case). If, in the first case, the currently considered belief state *bs* meets the new goal state, we abort the traversal of this path as it ends here. In the second case, if *bs* does not meet the new goal state, we have to take into account every possible new belief state that can follow right after *bs* and start planning from each of these new belief states in order to (possibly) retrieve new feasible paths that lead to *goal*. With this in mind, we first determine all actions $a \in app(bs)$ (retrieved from the set of actions *A*, cf. Definition 3.1.3) which were not planned in *bs* in the given process graph. For each of these actions we then determine the belief state $bs' = R(bs, a)$ and continue planning from bs'. If, during this planning, no belief state that meets *goal* is retrieved or no further action is applicable, the planning of the current path is aborted.

3.1.4.2.2 Removing a Goal State. Removing a goal state $goal \in GOALS$ (i.e., $GOALS' = GOALS \setminus \{goal\}$) implies that a termination criterion for the process is deleted. Therefore, each path in the given process graph that ends at a belief state meeting *goal* needs to be checked whether it can be extended by an existing planning technique so that it leads to one of the remaining goal states. If no goal state can be reached from

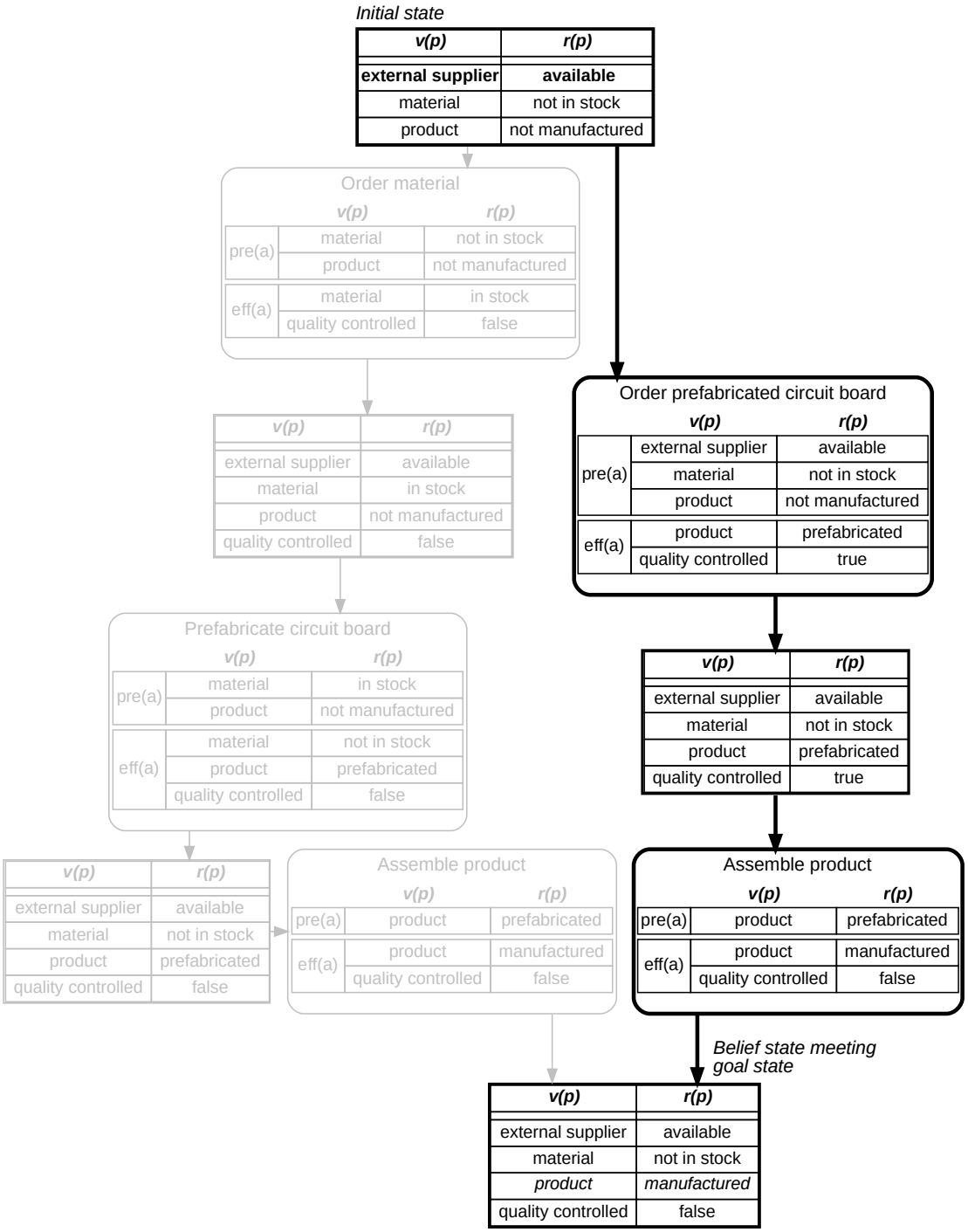

Figure 3.2: Process Graph after the Adaptation resulting from updating the Initial State

its last belief state (which formerly had met the now removed goal state *goal*), it is not considered in the adapted process graph. No other paths are affected by this change.

We therefore take into account each belief state *bs* of the given process graph that meets *goal* and try to reach one of the remaining goal states by planning (i.e., applying the transition function R entirely and computing all applicable actions and the belief states resulting from them), starting with each such *bs*. Thus, we first check each belief state *bs* that meets *goal* for the criteria of the remaining goal states. If *bs* meets the criteria of a remaining goal state, it is ensured that the paths which had ended at *goal* remain feasible in the adapted process graph. Else, the next planning step is executed: We determine the applicable actions in *bs* and construct the according resulting belief states by means of the transition function. Note that as soon as there are no actions applicable in the examined belief state and thus the planning step fails, the paths which had ended at goal cannot be extended to a feasible path and are therefore not considered in the adapted process graph.

3.1.4.2.3 Updating a Goal State. We separate the case of updating a goal state *goal* into two subcases. Since goal states serve as termination criteria, we distinguish between a strengthening update (i.e., making the conditions for meeting *goal* more severe) and a weakening update (i.e., making the conditions less severe). The updated goal state will be denoted by *goal'* (and thus $GOALS' = (GOALS \setminus \{goal\}) \cup \{goal'\}$). If a goal state is updated in any manner that is not included in the following two cases, we can represent this adaptation as a weakening update followed by a strengthening update.

Strengthening update. Strengthening the conditions of a goal state *goal* includes the addition of a belief state tuple to *goal* as well as changes to a belief state tuple $p \in goal$ limiting its restriction, formally replacing p by p' with $v(p)=v(p')$, $r(p) \neq \varnothing \neq r(p')$, $r(p) \neq r(p')$ and $r(p') \subset r(p)$ so that $goal' = (goal \setminus \{p\}) \cup \{p'\}$. When strengthening the conditions of *goal*, the set of (world) states that meet *goal'* is a proper subset of the set of states that meet *goal*, as these criteria are more severe. Thus, we proceed in a similar way to the case of removing a goal state (cf. Section 3.1.4.2.2): We start planning for each belief state *bs* meeting *goal* and each action that can be applied in *bs*, trying to reach one of the goal states from *GOALS'*.

Looking at the running example, a new compliance directive has come into force, requiring the company to integrate quality management as a documented and controlled task in the manufacturing process. Due to the new directive, it is required that the quality assurance is documented as an inherent part of the process. Therefore, the belief state tuple (*quality controlled,* {*true*}) is added to the goal state (bold and in italics). Thus, as seen in Figure 3.3, an action "External quality assurance" is now planned in the belief

state meeting the old goal state in order to meet the new, adapted goal state including the new belief state tuple.

Weakening update. This case covers removing a belief state tuple from *goal* as well as changes that extend the restriction of a belief state tuple $p \in goal$ (i.e., replacing p by p' with $v(p)=v(p')$, $r(p) \neq \varnothing \neq r(p')$, $r(p) \neq r(p')$ and $r(p) \subset r(p')$). Belief states meeting the goal state *goal* canonically meet *goal'*. Additionally, there are possibly further belief states meeting *goal'* which do not meet *goal*. Therefore, we align the approach to the case of adding the goal state *goal'* (cf. Section 3.1.4.2.1): We traverse all belief states in the process graph, check whether a belief state meets *goal'*, and try to retrieve new feasible paths to *goal'* by checking whether actions applicable in the belief states lead to *goal'* subsequently. In this way, feasible paths in the existing process graph may be shortened and new feasible paths may be constructed.

3.1.4.3 Changing (the Set of) Actions

As described in Definition 3.1.2, actions are triples consisting of the action name, the preconditions of the action and the effects of the action. According to CRUD, the requirement of an adaptation can arise from the addition of an action to the set of actions A, the removal of an action from A or the update of the preconditions or effects of an action in A (cf. Table 3.2).

3.1.4.3.1 Adding an Action. Let a be a new action so that $A'=A \cup \{a\}$. As a might be applicable in the given process graph, we need to check whether there exists a belief state bs in the given process graph such that a is applicable in bs. In such belief states we start planning by applying the transition function $R(bs,a)$. Further, there may exist paths $(bs_{\text{init}},a_1,\ldots,bs_{\text{k}})$ with $a_1,\ldots,a_{\text{k-1}} \in A$ that have not been feasible paths in the given process graph and with a being applicable in bs_{k}. In such belief states we also start planning by applying the transition function $R(bs_{\text{k}},a)$. Thereby, we possibly retrieve new feasible paths leading to a goal state.

Within the running example, the company decides to establish an own, internal quality assurance. This assurance, in difference to the external quality assurance contractor, is able to check the assembled product as well as (optionally) the internally prefabricated circuit board. As we see in Figure 3.4, an action "Internal quality assurance" (bold) is added to the process graph appropriately throughout the whole process.

3.1.4.3.2 Removing an Action. When removing an action a from A so that $A'=A \setminus \{a\}$, there can be no new feasible paths leading to a goal state. Further, each path in the

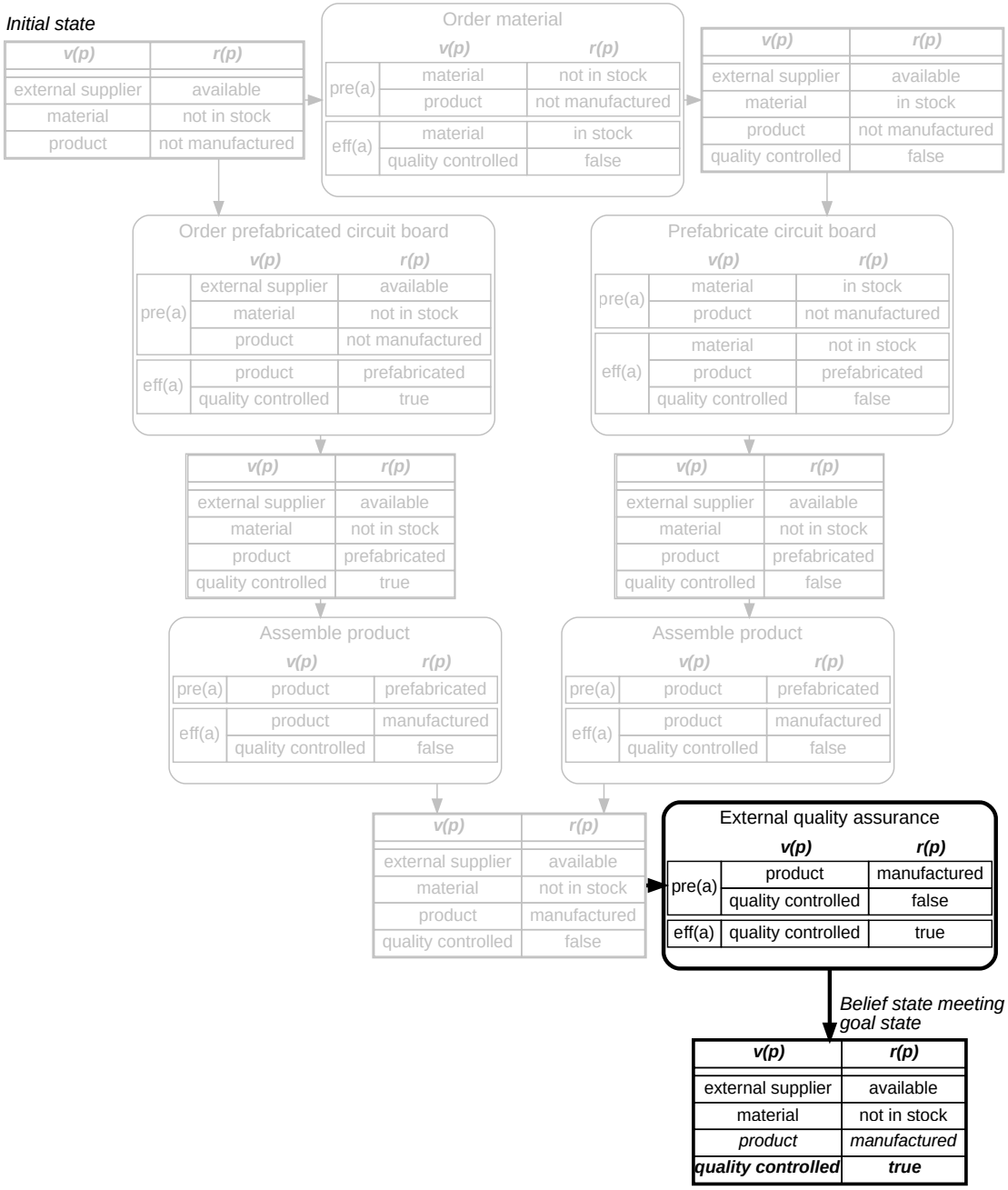

Figure 3.3: Process Graph after the Adaptation due to a strengthening Update of the Goal State

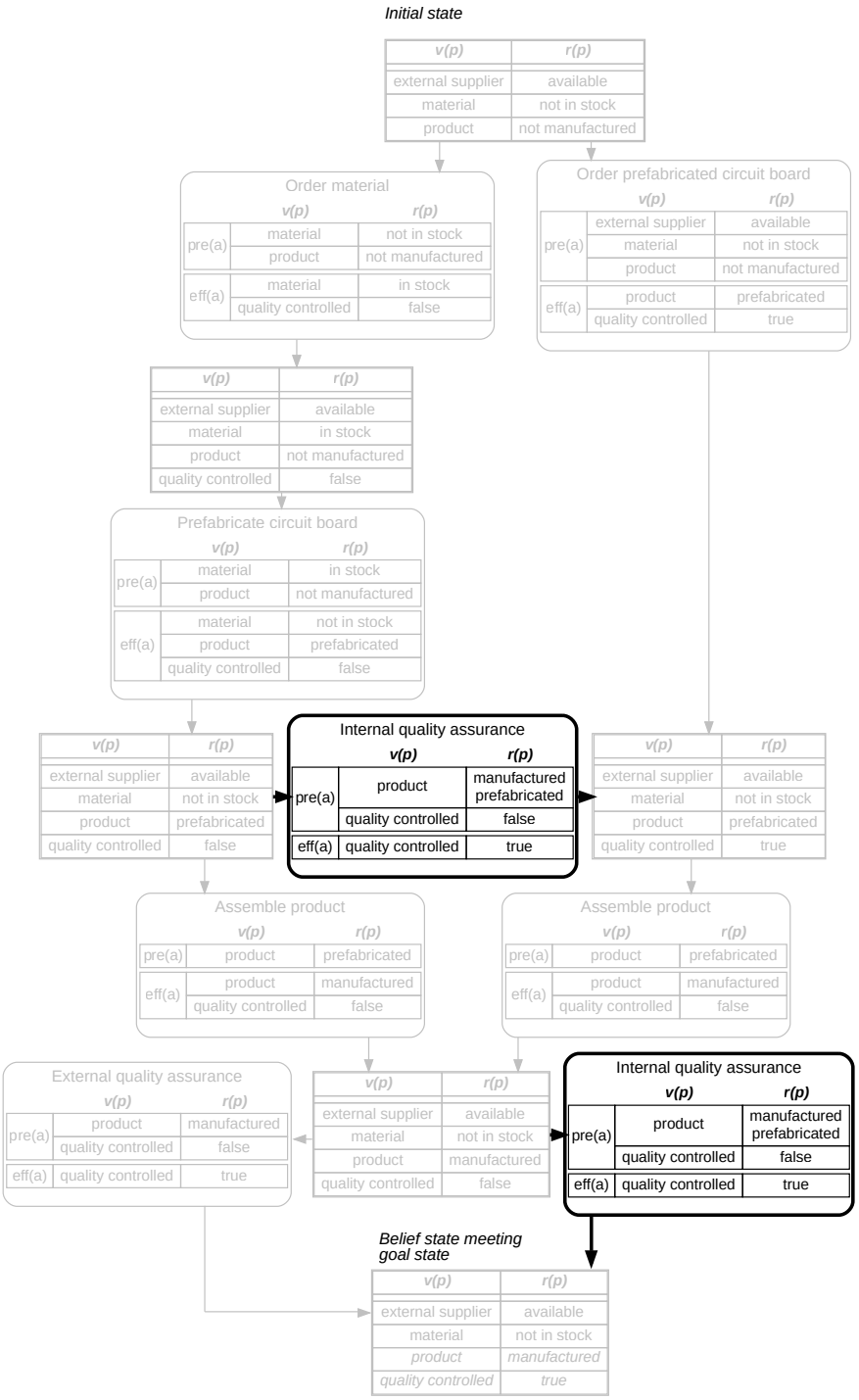

Figure 3.4: Process Graph after the Adaptation due to an added Action

given *process graph* containing a is not feasible in the adapted process graph and hence not retained. The paths not containing a are not affected at all and are retained.

3.1.4.3.3 Updating an Action. When updating an action a to a' we need to consider the case of updating the preconditions as well as the case of updating the effects of a. Updating the preconditions can be separated into the two subcases of strengthening and weakening updates since any update that is not covered by one of these two cases can be treated as performing an weakening update, followed by a strengthening update.

Strengthening update of the preconditions. When strengthening the preconditions of an action a (i.e., adding a belief state tuple p to $pre(a)$ or updating p to p' so that $v(p)=v(p')$, $r(p)\neq\varnothing\neq r(p')$, $r(p)\neq r(p')$, $r(p')\subset r(p)$ and $pre(a')=(pre(a)\setminus\{p\})\cup\{p'\}$), only a subset of the belief states of the given process graph in which a was applicable also fulfills the requirements for the applicability of a'. Hence, we need to check for each belief state bs in which a was applicable whether a' is still applicable. If this is not the case, we do not consider the paths containing a' in the adapted process graph (cf. case of removing an action, Section 3.1.4.3.2). On the other hand, if a' is still applicable in bs, the belief state bs_1 that results from $R(bs,a)$ may differ from the belief state $bs_1{}'$ resulting from $R(bs,a')$ (cf. Definition 3.1.2). In this case, it is possible that the sets $app(bs_1)$ and $app(bs_1{}')$ do not coincide. We then proceed analogously as we did when treating the case of updating the initial state (cf. Section 3.1.4.1) with $bs_1{}'$ taking the role of the updated state to bs_1.

Weakening update of the preconditions. When weakening the preconditions of an action a (i.e., removing a belief state tuple from $pre(a)$ or updating p to p' so that $v(p)=v(p')$, $r(p)\neq\varnothing\neq r(p')$, $r(p)\neq r(p')$, $r(p)\subset r(p')$ and $pre(a')=(pre(a)\setminus\{p\})\cup\{p'\}$) it is possible that a' becomes applicable in additional belief states in which a has not been applicable. We therefore check each belief state bs of the given process graph with $a\notin app(bs)$ in regard to $a'\in app(bs)$. If, indeed, $a'\in app(bs)$ holds, we apply a planning approach in accordance to the case of adding a new action (cf. Section 3.1.4.3.1). Further, there may exist paths $(bs_{\text{init}},a_1,\ldots,bs_{\text{k}})$ with $a_1,\ldots,a_{\text{k-1}}\in A$ that have not been feasible paths in the given process graph and with $a\notin app(bs_{\text{k}})$, but $a'\in app(bs_{\text{k}})$. In such belief states we also start planning by applying the transition function $R(bs_{\text{k}},a')$. Thereby, we may retrieve new feasible paths leading to a goal state. Additionally, the same situation as in the preceding paragraph ($R(bs,a)\neq R(bs,a')$) can arise and is handled in the same manner as above (cf. Section 3.1.4.1).

Updating the effects. Finally, when updating the effects of an action a with respect to a single belief state tuple, we consider each belief state bs of the given process graph in which a is applicable. Due to the changed effects, once again, we may encounter

Type of atomic change (cf. Table 3.2)	Main enhancements: Reuse of applicability information	Main enhancements: Reuse of existing subgraphs	Main enhancements: Planning based on existing process graph
Update initial state	✔	✔	✖
Add goal state	✖	✖	✔
Update goal state	✖	✖	✔
Remove goal state	✖	✖	✔
Add action	✖	✖	✔
Update action	✔	✔	✔
Remove action	✖	✖	✔

Table 3.3: Enhancements over Existing Planning Approaches

the situation in which $R(bs,a) \neq R(bs,a')$ holds, which is handled as above (cf. Section 3.1.4.1).

3.1.4.4 Summary of the Approach

In the Sections 3.1.4.1-3.1.4.3 it was shown how to adapt a process graph to each of the atomic changes specified in Table 3.2. Table 3.3 summarizes the main enhancements with regard to existing methods from automated planning which do not reuse any results from previous planning runs.

As all adaptations can be realized as a sequence of these atomic changes, this means that a full-featured approach for the adaptation of process models has been developed. We discuss this by means of our running example:

In order to enter new markets, a new manufacturing facility is built by the electrical engineering company. In this new facility the manufacturing process from above (cf. Figure 3.4) is planned to be applied, however it needs to be adapted. To reach a broad market coverage, a second production line for the prefabrication of circuit boards consisting of two machines has to be added. Additionally, analyses show that a new packaging is needed for this market and hence, product packing is planned to be included into the manufacturing process. As the external quality assurance contractor does not operate in this market, it is planned to exclusively handle quality assurance at

the facility. Furthermore, local regulatory requirements demand the quality assurance for prefabricated circuit boards to be mandatory.

From this description the corresponding atomic changes can be inferred directly. First, a second production line is incorporated into our process graph by adding the actions "Prefabricate circuit board on machine 1" and "Prefabricate circuit board on machine 2". These actions have preconditions and effects similar to the action "Prefabricate circuit board" with the only difference being the belief state tuple $(product, \{in\ prefabrication\})$ which is needed as these actions have to be put in sequence. Second, product packing is enabled by adding the action "Packing product" and updating the goal state to contain the belief state tuple $(product, \{packed\})$. With these atomic changes, the ability as well as the necessity for a manufactured product to be packed is given. Third, to meet the business changes regarding quality assurance, the action "External quality assurance" is deleted. Additionally, the belief state tuple $(quality\ controlled, \{true\})$ is added to the preconditions of the action "Assemble product" to comply with legal requirements. In this way prefabricated circuit boards cannot be processed without having their quality checked. The resulting process graph is shown in Figure 3.5.

3.1.5 Evaluation

We assessed our approach based on evaluation criteria stated in literature (Prat et al., 2015). In particular, the following Table 3.4 shows the four analyzed evaluation questions, the respective evaluation criterion, the way for analysis and the Section or Appendix in which a detailed description is given (please note that parts of the evaluation were moved to the Appendix due to their length).

More precisely, the evaluation criterion regarding correctness and completeness (E1) was proved (cf. Appendix 7.4.1 for a detailed discussion), which shows that our approach can indeed adapt process models such that the resulting process models are correct and complete. This finding supports contribution ❷. To evaluate (E3), the approach was used in a real-world scenario for the adaptation of existing process models to needs for change in advance in an automated manner (cf. ❶). Further, it was shown by means of a simulation experiment as well as an algorithmic complexity analysis (E4) that the presented approach provides advantages in performance and computational complexity compared to planning the process graphs from scratch.

3.1.5.1 Evaluation of (E2) Technical Feasibility

We implemented our approach for the automated adaptation of process models in a software prototype. An existing Java implementation of a planning technique (cf. Bertoli et al., 2006; Heinrich et al., 2015c) for nondeterministic state transition systems able to

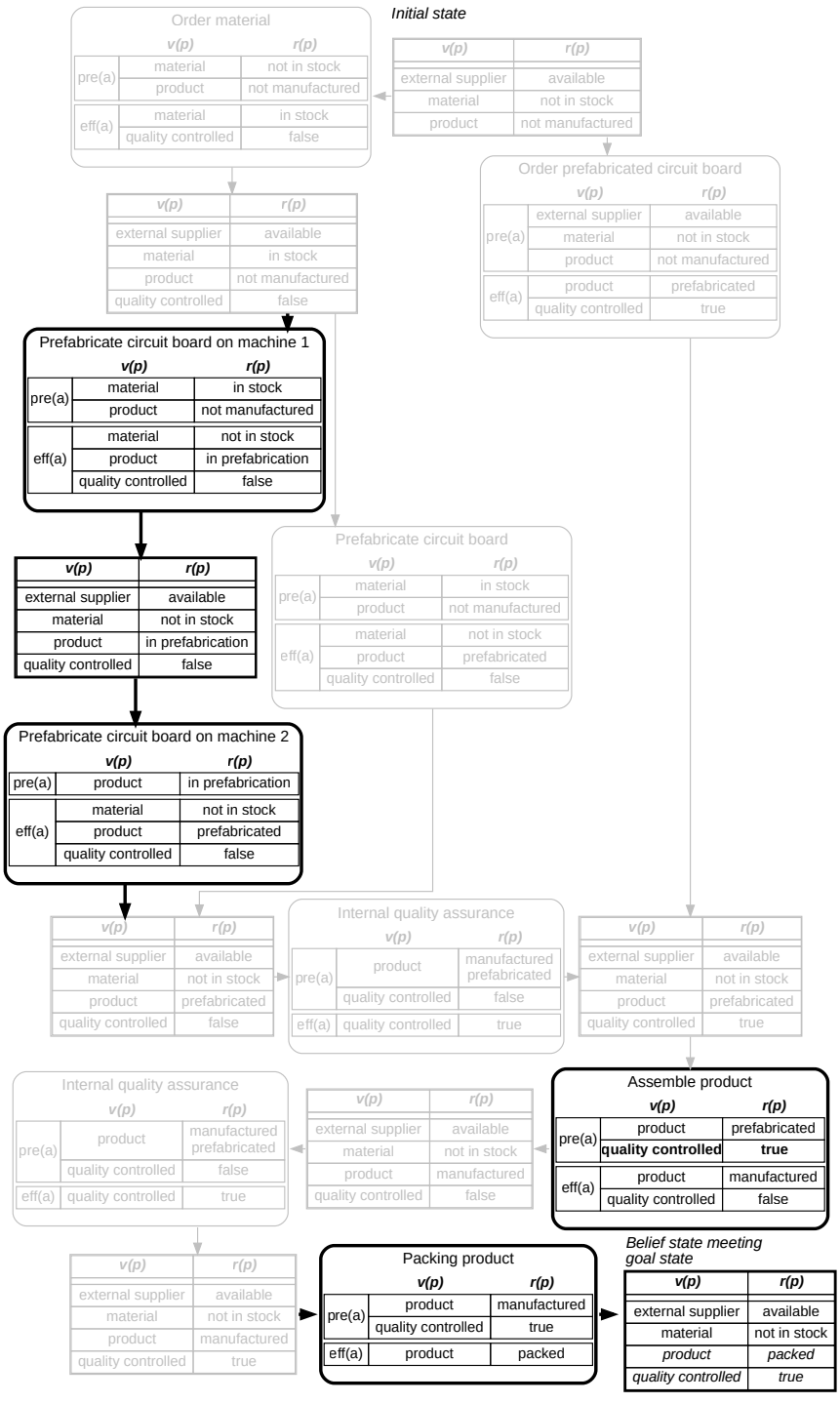

Figure 3.5: Process Graph after the Adaptation due to *multiple Atomic Changes*

	Evaluation question	Evaluation criterion	Way for analysis	Section / Appendix
(E1)	Does the proposed approach terminate and construct correct and complete adapted process models?	Correctness and Completeness	Mathematical proofs	7.4.1
(E2)	Can the approach be instantiated?	Technical feasibility	Prototypical software implementation, Pseudocode	3.1.5.1, 7.4.2
(E3)	Can the approach be successfully applied in a real-world scenario?	Operational feasibility	Application to manufacturing process of an engineering company	3.1.5.2
(E4)	Does the approach provide performance advantages compared to planning from scratch?	Computational complexity	Simulation experiment based on 12 real-world processes, Algorithmic complexity analysis	3.1.5.3, 7.4.3

Table 3.4: Overview of Evaluation

construct process graphs (cf. Definition 3.1.5) served as a basis. The existing implementation allows to specify planning problems by means of XML files. We added the functionality to also specify changes to a process graph via XML files. In particular, using a set of XML files (one file specifying the initial regular planning problem and another file specifying changes), a process graph and one to possibly many subsequent changes can be specified. We further integrated the previously presented approach (cf. Sections 3.1.4.1 to 3.1.4.3) in the implemented planning technique. Persons other than the programmers validated the source code via structured walkthroughs. Moreover, the validity of this extended prototype was ensured by carrying out structured tests using the JUnit framework. The pseudocode of our implementation can be found in Section 7.4.2 in the Appendix. This supports the technical feasibility (E2) of our approach.

3.1.5.2 Evaluation of (E3) Operational Feasibility

In order to evaluate whether the approach can adapt process models to needs for change in advance (contribution ❶) and thus operational feasibility, we conducted a field experiment by applying our approach to a manufacturing process of a European electrical engineering company. To do so, we interviewed the manager of the manufacturing department about a process that was subject to several adaptations in recent history. Based on a first interview, the annotations of actions, initial state and goal states of the original process (that was in place before these adaptations) could be prepared. In a second meeting, we reviewed them together with the staff to validate that their specification was accurate. Thereafter, a detailed process graph, depicting the existing process (consisting of 27 actions, 20 belief states and 48 paths; cf. Table 3.5) could be planned by means of the aforementioned Java implementation. The running example used within this paper is a simplified excerpt of this graph. During further meetings, the manager provided us with information about the aforementioned needs for change to this process that took place in recent history. Please note that despite the changes to the processes had occurred in the past, our approach still adapted to a need for change in advance in this setting because only the need for change was used as input and the actually conducted processes and resulting changes to it just served as reference for comparison to our adaptation result. To assess the operational feasibility of our approach, we analyzed the following three questions necessary for a successful application of our approach in this real-world scenario:

(E3.1) Using our approach, is it possible to adapt the process graph to the needs for change stated by the manager?

(E3.2) Do the adapted process graphs represent the actually used process models?

(E3.3) Are the adapted process graphs assessed as correct and complete by the staff?

Ad (E3.1): Based upon the information given by the manager, we were able to determine atomic changes and specify them in terms of the aforementioned XML files. Subsequently, we adapted the process graph by means of our prototype in the order the changes occurred in reality (cf. Table 3.5). The first adaptation resulted from the demand to consider the situation of prefabricated packaging being in stock (an update of the initial state). This adaptation resulted in 15 old and 5 updated belief states in the adapted process graph. Thereafter, based on the adapted process graph we addressed the second need for change and so on. Overall, with respect to Table3.1, the changes "updating the initial state", "adding an action", "removing an action" and "updating a goal state" were addressed. All needs for change could be represented and addressed.

Ad (E3.2): In order to compare the adapted process graphs with the respective actual processes conducted after each change in the company, we scheduled a further meeting with the staff of the engineering company. After the first adaptation (cf. third row in Table 3.5), we presented the resulting process graph to the staff and discussed the differences with them. However, we observed that the staff had some problems comprehending this graph. Therefore, for the subsequent adaptations (cf. rows four to eight in Table 3.5), we visually simplified the adapted process graphs so that they became more understandable for the staff. In detail, we removed the belief states from the versions presented to the company and omitted the preconditions and effects of the actions represented in the graphs so that their layout was similar to UML activity diagrams. Still, the complete process model was presented. Then, we discussed these graphs with the manager and employees of the manufacturing department. Particularly, for each need for change (cf. Table 3.5), we elaborated the differences between the graph before adaptation and the adapted graph in detail. Thereby, we asked the staff whether the adapted graphs represent the processes as they had actually been conducted in the company as soon as the according change took place. The staff confirmed this for every case.

Ad (E3.3): The staff further assessed the paths in the adapted process graphs to be correct. We also asked whether the adapted graphs neglected any feasible paths. Here, the staff validated that the graphs contained all paths actually used in the company and that no feasible paths not represented in the adapted process graphs were known. Please note that while the number of paths to check was high in some cases, most paths just contained the same actions in different order, making a manual verification possible.

To conclude, the (E3) operational feasibility of our approach could be supported in this real-world scenario. Our approach was able to adapt process models to needs for change in advance in an automated manner (cf. ❶). However, we observed that the resulting process graphs are not yet easy to comprehend. This issue may be solved

Description of the needs for change	Type of atomic change (cf. Table 3.2)	feasible paths	actions	Number of … old belief states	new	updated	belief states (in total)
			… in the process graph after the adaptation				
Original process graph (starting point)	-	48	27	-	-	-	20
Considering the situation that prefabricated packaging is in stock	Updating the initial state	56	28	15	0	5	20
Considering the situation that there is an external supplier for circuit boards	Updating the initial state	76	45	7	12	13	32
Preproduction of spare parts can now be done by the company itself	Adding an action	80	55	32	0	9	41
A quality assurance department is set up internally	Adding an action	460	76	41	0	9	50
A production machine for circuit boards is sold	Removing an action	268	70	47	0	0	47
Testing the functionality of the product at the installation site is required	Updating a goal state	2,412	136	47	42	0	89

Table 3.5: Adaptations performed in the case of the Engineering Company

by using approaches to construct control flow patterns based on our adapted process graph (Heinrich et al., 2015b; Heinrich et al., 2016; Meyer et al., 2006). For assessing (E3), however, it was sufficient to describe the process graphs to the staff and discuss the changes of these graphs in detail.

3.1.5.3 Evaluation of (E4) Performance

We additionally conducted an analysis to evaluate computational complexity as well as a simulation experiment to analyze the difference in performance of our approach compared to planning from scratch[1].

Overall, the complexity analysis (cf. Section 7.4.3 in the Appendix) shows that our approach provides considerable advantages in computational complexity compared to planning process graphs from scratch.

Regarding the simulation experiment, we focus on atomic changes to provide transparent results. However, please note that all possible adaptations can be realized as a sequence of these atomic changes. For our analysis, we measured absolute runtimes as well as each ratio, which means, the absolute runtime for adaptation divided by the corresponding absolute runtime for planning from scratch for all possible types of atomic change (cf. Table 3.2). To do so, we used adaptation cases based upon 12 existing real-world process graphs of different companies from the application contexts *Project Management*, *Insurance Management*, *Loan Management* and *Private Banking* (cf. Table 3.6). The process graphs consist of 17 to 8,267 actions and 15 to 2,693 belief states and contain numeric domains as well as discrete domains in their belief state tuples. There are two process graphs that stand out regarding the number of feasible paths (*Selling an insurance contract* and *Contracting wealth management customer*). In these two cases, large parts could be conducted in parallel to other large parts in the same process. Process graphs do not contain control flow patterns and paths represent sequences of subsequent belief states and actions. Hence, these two process graphs represent large processes of complete value chains (including back office) in which many actions can be executed in multiple different orders. This results in process graphs comprising a vast number of feasible paths, each of which represents a different possible order of actions. Yet, the number of distinct state variables remains rather small because these central business state variables can have many different values and auxiliary state variables, which were not counted here. We deliberately analyzed these two process graphs to show the feasibility of our approach regarding large process graphs.

In a first step, we defined random adaptation cases for each type of atomic change

[1]We executed the prototype on a Dell Latitude Notebook with an Intel Core i7-2640M, 2.80 GHz, 8GB RAM running on Windows 8.1 (Version 6.3.9600) 64 bit and Java 1.8.0 (build 1.8.0.-b132) 64 bit.

Application Context	Process description	Number of actions in the given process graph	Number of belief states in the given process graph	Number of feasible paths in the given process graph	Number of distinct state variables
Project Management	Preparing and coordinating project profile	17	15	6	4
Project Management	Specifying project resources	25	18	36	7
Project Management	Allocating project resources	26	22	24	6
Project Management	Preparing the project report for the board	38	25	252	5
Insurance Management	Administrating customer and product database	43	38	52	7
Insurance Management	Handling insured events	54	44	60	6
Insurance Management	Selling an insurance contract	8,267	2,693	97,501,324,491	9
Loan Management	Analyzing credit rating	40	31	197	6
Loan Management	Selling mortgage loans	57	43	216	8
Loan Management	Settling mortgage loans	122	69	6,572	11
Private Banking	Contracting wealth management customer	278	189	1,244,416	27
Private Banking	Order management	82	74	32	19

Table 3.6: Key Properties of used real-world Processes

based on the belief state variables of each of the 12 process graphs. Each of these generated adaptation cases was then (automatically) validated with regard to its validity. For instance, a goal state could only be removed if there was at least one goal state remaining after the adaptation. Invalid adaptation cases remained unconsidered. We randomly generated 1,500 valid adaptation cases for each type of atomic change over all 12 process graphs. For each case, the specification of the process graph, the specification of the adaptation and the specification for planning the adapted process graph from scratch were automatically prepared in terms of XML files that could be imported in our prototypical implementation.

We applied our approach to these adaptation cases and automatically verified that adapting the given process graphs and planning the adapted process graphs from scratch resulted in exactly the same process model in each case. Then, we compared the runtime required for adapting the given process graphs with the runtime required for planning the graphs from scratch. Both runtimes do not take into account the time required to generate the XML files representing the adaptation cases. The results of this runtime comparison can be seen in Figure 3.6.

We observed that the required runtime for adapting the existing process graphs by the prototype is lower than for planning the graphs from scratch for each type of atomic change. The left part of Figure 3.6 shows the mean of the required absolute runtime for adapting the process graphs (first line in each cell) as well as the mean percentage ratio (absolute runtime for adaptation divided by absolute runtime for planning from scratch; second line in each cell), which varies between 3.68% and 10.52%, depending on the type of atomic change. To give an example of a process, independent of the type of atomic change our approach takes on average 0.35 seconds for adapting the graph of the process *Selling an insurance contract*. In contrast, planning from scratch would on average take about 10 seconds, which leads to an average time saving of 96.4%. The right part of Figure 3.6 shows box plots of these percentage ratios. The left and right ends of the boxes are the first and third quartiles, and the bands inside the boxes are the medians. The whiskers (i.e., the horizontal lines outside of the boxes) include all values within 1.5*interquartile range.

These results of the simulation experiment show that while for just a few adaptation cases, adapting instead of planning from scratch provided only negligible or even non-existent runtime advantages, for most adaptation cases, the runtime advantage was considerable. Further, the results support that using our approach provides considerable performance advantages, even if a sequence of changes has to be addressed. To analyze the significance of our findings, we further conducted a one-tailed paired t-test. This kind of test was chosen because we aimed to analyze a paired sample dataset (runtimes for adapting versus planning from scratch). There was a significant differ-

	Add	Update	Remove
Initial state	—	7.73ms 9.05%	—
Goal states	3.52ms 3.80%	6.07ms 7.32%	0.58ms 3.68%
Actions	11.67ms 10.52%	18.68ms 7.80%	2.87ms 5.71%

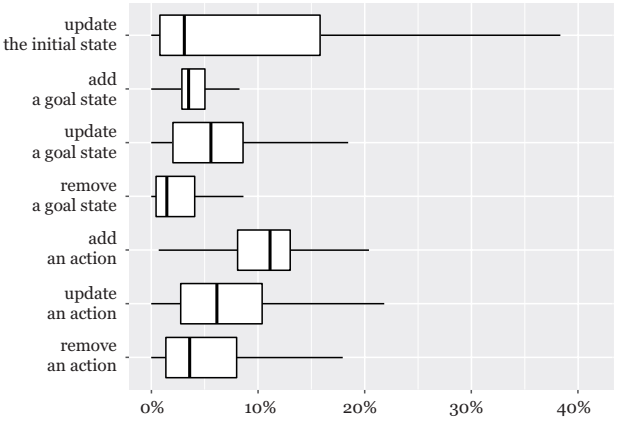

Figure 3.6: Evaluation Results by means of a Prototypical Implementation

ence in the runtimes according to the results of the t-test: *degrees of freedom*=139,190; *t-value*=88.685, *p-value*=2.2e-16. The *p*-value was consistently less than or equal to 2.2e-16 across all types of atomic changes. These significant results support the thesis that our approach is faster than planning the adapted process graphs from scratch. This is especially advantageous when working with graphical process modeling tools. In such tools, modelers can – *in real time* – conduct a sequence of changes in order to, step by step, adapt a given process model to needs for change. The previously mentioned example of adapting *Selling an insurance contract* underlines this argument. Here, for instance, using the presented approach would on average result in a 0.35 second waiting time after each entered change instead of 10 seconds when planning from scratch. Thus, our approach enables modelers to work much more comfortably compared to using existing approaches.

3.1.6 Conclusion, Limitations and Future Work

In this paper, we presented a novel approach for the automated adaptation of process models that – in contrast to existing works – constructs correct and complete process models (cf. contribution ❷). This approach can be used to adapt existing process models to needs for change in advance in an automated manner (cf. contribution ❶). We mathematically verified our approach, showed its technical feasibility by means of a prototypical software implementation and its operational feasibility by applying the approach to a real-world process in a field experiment. Additionally, we conducted a complexity analysis which shows that our approach provides considerable advantages in regard to computational complexity compared to planning the process graph from

scratch. Moreover, we analyzed the performance of our approach in a simulation experiment.

Our research possesses some limitations, which should be addressed in future work. First, our approach adapts process graphs without cycles (cf. Definition 3.1.5). However, the (sub) paths within a cycle could be analyzed once and separately using the approach presented in Section 3.1.4. In this way, process graphs containing arbitrary cycles could be adapted as well. Second, we evaluated the operational feasibility of our approach by applying it to a single real-world scenario in an experimental setting. Thus, the presented approach should be further evaluated in a broader context. In particular, a larger number of field experiments in different industry sectors should be conducted to verify the operational feasibility.

Further, the runtime of adapting a process model to a (very) large number of changes may be slower than planning a process model from scratch. Although we have already provided some insights on this topic by means of the simulation experiment presented above, additional work needs to be done. For instance, an estimation regarding the expected runtime of adapting a given process model compared to the expected runtime of planning it from scratch, based on the needs for change, would be useful. This could provide fruitful insights for deciding whether to use our approach or to plan the process model from scratch, given a large number of needs for change. Further, we aim to construct complete adapted graphs (cf. contribution ❷) and thus do not focus on a heuristic approach in this paper. However, the runtime for adapting process graphs may additionally be reduced further by means of a heuristic approach in case a process model with all feasible paths is not necessary.

Moreover, research work has to be done in order to transfer our approach to related fields. While a larger number of approaches in (web) service composition utilizes concepts of automated planning (including notions of "states" and "actions" with "preconditions" and "effects", cf. Fan et al., 2018; Montarnal et al., 2018; Rao et al., 2005) recent research has shown how to apply similar concepts to the field of process mining as well cf., e.g., Montarnal et al., 2018; Song et al., 2016. To extensively evaluate the feasibility of our approach in these fields it needs to be implemented in the according toolsets and applied to different real-world scenarios. We hope that our work will open doors for further research in this exciting area.

3.1.7 References

Afflerbach, P., Kastner, G., Krause, F., and Röglinger, M. (2014). "The Business Value of Process Flexibility." In: *Business & Information Systems Engineering* 6.4, pp. 203–214. ISSN: 1867-0202. DOI: 10.1007/s12599-014-0333-5.

Alférez, G. H., Pelechano, V., Mazo, R., Salinesi, C., and Diaz, D. (May 2014). "Dynamic adaptation of service compositions with variability models." In: *Journal of Systems and Software* 91, pp. 24–47.

Augusto, A., Conforti, R., Dumas, M., Rosa, M. L., Maggi, F. M., Marrella, A., Mecella, M., and Soo, A. (Apr. 2019). "Automated Discovery of Process Models from Event Logs: Review and Benchmark." In: *IEEE Transactions on Knowledge and Data Engineering* 31.4, pp. 686–705. ISSN: 2326-3865. DOI: 10.1109/TKDE.2018.2841877.

Barba, I., Del Valle, C., Weber, B., and Jiménez-Ramírez, A. (2013a). "Automatic generation of optimized business process models from constraint-based specifications." In: *International Journal of Cooperative Information Systems* 22.02, p. 1350009. DOI: 10.1142/S0218843013500093.

Barba, I., Weber, B., Del Valle, C., and Jiménez-Ramírez, A. (2013b). "User recommendations for the optimized execution of business processes." In: *Data & Knowledge Engineering* 86, pp. 61–84. DOI: 10.1016/j.datak.2013.01.004.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bider, I. (2005). "Masking flexibility behind rigidity: Notes on how much flexibility people are willing to cope with." In: *Proceedings of the CAiSE* 5, pp. 7–18.

Bortlik, M., Heinrich, B., and Mayer, M. (Oct. 2018). "Multi User Context-Aware Service Selection for Mobile Environments." In: *Business & Information Systems Engineering* 60.5, pp. 415–430. DOI: 10.1007/s12599-018-0552-2.

Bose, R. J. C., van der Aalst, W. M. P., Zliobaite, I., and Pechenizkiy, M. (2014). "Dealing with concept drifts in process mining." In: *IEEE transactions on neural networks and learning systems* 25.1, pp. 154–171. ISSN: 2162-2388. DOI: 10.1109/TNNLS.2013.2278313.

Bucchiarone, A., Pistore, M., Raik, H., and Kazhamiakin, R. (2011). "Adaptation of service-based business processes by context-aware replanning." In: *Service-Oriented Computing and Applications (SOCA), 2011 IEEE International Conference on*. IEEE, pp. 1–8. ISBN: 1467303186.

Canfora, G., Di Penta, M., Esposito, R., and Villani, M. L. (2005). "An Approach for QoS-aware Service Composition based on Genetic Algorithms." In: *Proceedings of the 7th annual conference on Genetic and evolutionary computation (GECCO '05)*, pp. 1069–1075.

Chafle, G., Dasgupta, K., Kumar, A., Mittal, S., and Srivastava, B. (2006). "Adaptation in Web Service Composition and Execution." In: *Proceedings of the 2006 IEEE International Conference on Web Services (ICWS'06)*, pp. 549–557.

Chen, L., Li, X., and Yang, Q. (2012). "Continuous process improvement based on adaptive workflow mining technique." In: *Journal of Computational Information Systems* 8.7, pp. 2891–2898.

Cognini, R., Corradini, F., Gnesi, S., Polini, A., and Re, B. (Apr. 2018). "Business process flexibility - a systematic literature review with a software systems perspective." In: *Information Systems Frontiers* 20.2, pp. 343–371. ISSN: 1387-3326. DOI: 10.1007/s10796-016-9678-2.

De Leoni, M. and Marrella, A. (2017). "Aligning Real Process Executions and Prescriptive Process Models through Automated Planning." In: *Expert Systems with Applications* 82, pp. 162–183. ISSN: 09574174. DOI: 10.1016/j.eswa.2017.03.047.

Döhring, M., Reijers, H. A., and Smirnov, S. (2014). "Configuration vs. adaptation for business process variant maintenance: An empirical study." In: *Information Systems* 39, pp. 108–133. ISSN: 0306-4379. DOI: 10.1016/j.is.2013.06.002.

Eisenbarth, T. (2013). *Semantic Process Models: Transformation, Adaptation, Resource Consideration*. Augsburg: Universität Augsburg.

Eisenbarth, T., Lautenbacher, F., and Bauer, B. (2011). "Adaptation of Process Models – A Semantic-based Approach." In: *Journal of Research and Practice in Information Technology* 43.1, pp. 5–23.

Ellis, C., Keddara, K., and Rozenberg, G. (1995). "Dynamic change within workflow systems." In: *Proceedings of the conference on Organizational computing system (COCS '95)*. Ed. by N. Comstock and C. Ellis, pp. 10–21. DOI: 10.1145/224019.224021.

Fahland, D., Favre, C., Koehler, J., Lohmann, N., Völzer, H., and Wolf, K. (2011). "Analysis on demand: Instantaneous soundness checking of industrial business process models." In: *Data & Knowledge Engineering* 70.5, pp. 448–466. DOI: 10.1016/j.datak.2011.01.004.

Fahland, D. and van der Aalst, W. M. P. (2012). "Repairing Process Models to Reflect Reality." In: *Business Process Management* 7481, pp. 229–245.

Fan, S.-L., Yang, Y.-B., and Wang, X.-X. (June 2018). "Efficient Web Service Composition via Knapsack-Variant Algorithm." In: *Proceedings of the 15th International Conference on Services Computing (SCC 2018)*. Ed. by J. E. Ferreira, G. Spanoudakis, Y. Ma, and L.-J. Zhang. Seattle, WA, USA: Springer International Publishing, pp. 51–66. ISBN: 978-3-319-94376-3.

Forstner, E., Kamprath, N., and Röglinger, M. (2014). "Capability development with process maturity models – Decision framework and economic analysis." In: *Journal*

of Decision Systems 23.2, pp. 127–150. ISSN: 1246-0125. DOI: 10.1080/12460125.2014.865310.

Gambini, M., La Rosa, M., Migliorini, S., and ter Hofstede, A. H. M. (2011). "Automated error correction of business process models." In: *Business Process Management*. Ed. by S. Rinderle-Ma, F. Tourmani, and K. Wolf. Springer, pp. 148–165.

García-Bañuelos, L., van Beest, N., Dumas, M., La Rosa, M., and Mertens, W. (Mar. 2018). "Complete and Interpretable Conformance Checking of Business Processes." In: *IEEE Transactions on Software Engineering* 44.3, pp. 262–290. ISSN: 2326-3881. DOI: 10.1109/TSE.2017.2668418.

Garrido, A., Guzman, C., and Onaindia, E. (2010). "Anytime plan-adaptation for continuous planning." In: *PlanSIG'10*, pp. 62–69.

Gerevini, A. and Serina, I. (Apr. 2000). "Fast Plan Adaptation through Planning Graphs: Local and Systematic Search Techniques." In: *Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems*. AIPS'00. Breckenridge, CO, USA: AAAI Press, pp. 112–121. ISBN: 1577351118.

Gerevini, A. E., Saetti, A., and Serina, I. (2012). "Case-based Planning for Problems with Real-valued Fluents: Kernel Functions for Effective Plan Retrieval." In: *Frontiers in Artificial Intelligence and Applications* 242.ECAI 2012, pp. 348–353.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Ghallab, M., Nau, D. S., and Traverso, P. (2016). *Automated Planning and Acting*. New York, NY: Cambridge University Press. ISBN: 978-1107037274.

Hallerbach, A., Bauer, T., and Reichert, M. (2010). "Capturing variability in business process models: the Provop approach." In: *Journal of Software Maintenance and Evolution: Research and Practice* 22.6–7, pp. 519–546. ISSN: 1532-0618.

Hammer, M. (2015). "What is Business Process Management?" In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 3–16. ISBN: 978-3-642-45099-0. DOI: 10.1007/978-3-642-45100-3_1.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems (ICIS)*. Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Heinrich, B. and Mayer, M. (2018a). "Service selection in mobile environments: considering multiple users and context-awareness." In: *Journal of Decision Systems* 27.2, pp. 92–122. DOI: 10.1080/12460125.2018.1513223.

Heinrich, B., Schiller, A., and Schön, D. (2018b). "The cooperation of multiple actors within process models: an automated planning approach." In: *Journal of Decision Systems* 27.4, pp. 238–274. DOI: 10.1080/12460125.2019.1600894. URL: https://epub.uni-regensburg.de/40166/.

Heinrich, B. and Schön, D. (2015c). "Automated Planning of context-aware Process Models." In: *Proceedings of the 23rd European Conference on Information Systems (ECIS)*. Ed. by J. Becker, J. vom Brocke, and M. de Marco. Münster, Germany, Paper 75.

Heinrich, B. and Schön, D. (2016). "Automated Planning of Process Models: The Construction of Simple Merges." In: *Proceedings of the 24rd European Conference on Information Systems (ECIS)*. Istanbul, Turkey.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik (MKWI)*. Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hoffmann, J., Weber, I., and Kraft, F. M. (2009). "Planning@ sap: An application in business process management." In: *Proceedings of the 2nd International Scheduling and Planning Applications woRKshop (SPARK'09)*.

Hoffmann, J., Weber, I., and Kraft, F. M. (July 2012). "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management." In: *Journal of Artificial Intelligence Research* 44.1, pp. 587–632. DOI: 10.1613/jair.3636.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference on Advanced Information Systems Engineering (CAiSE)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

Hull, R. and Motahari Nezhad, H. R. (2016). "Rethinking BPM in a Cognitive World: Transforming How We Learn and Perform Business Processes." In: *Business Process Management*. Ed. by M. La Rosa, P. Loos, and O. Pastor. Cham: Springer International Publishing, pp. 3–19. ISBN: 978-3-319-45348-4.

IEEE Task Force on Process Mining (2012). "Process Mining Manifesto." In: *Business Process Management Workshops*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, F. Daniel, K. Barkaoui, and S. Dustdar. Vol. 99. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 169–194. ISBN: 978-3-642-28107-5. DOI: 10.1007/978-3-642-28108-2_19.

Jiménez-Ramírez, A., Barba, I., Del Valle, C., and Weber, B. (2013). "Generating Multi-objective Optimized Business Process Enactment Plans." In: *Advanced Information Systems Engineering*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. Salinesi, M. C. Norrie, and Ó. Pastor. Vol. 7908. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 99–115. ISBN: 978-3-642-38708-1. DOI: 10.1007/978-3-642-38709-8_7.

Kalenkova, A. A., van der Aalst, W. M. P., Lomazova, I. A., and Rubin, V. A. (Oct. 2017). "Process Mining Using BPMN: Relating Event Logs and Process Models // Process mining using BPMN: Relating event logs and process models." In: *Software and Systems Modeling* 16.4, pp. 1019–1048. DOI: 10.1007/s10270-015-0502-0.

Kambhampati, S. (1997). "Refinement Planning as a Unifying Framework for Plan Synthesis." In: *AI MAGAZINE* 18.2, pp. 67–98.

Katzmarzik, A., Henneberger, M., and Buhl, H. U. (2012). "Interdependencies between automation and sourcing of business processes." In: *Journal of Decision Systems* 21.4, pp. 331–352. ISSN: 1246-0125. DOI: 10.1080/12460125.2012.749755.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security (IJCSIS)* 7.2, pp. 147–152.

La Rosa, M., van der Aalst, W. M. P., Dumas, M., and Milani, F. P. (Mar. 2017). "Business Process Variability Modeling: A Survey." In: *ACM Comput. Surv.* 50.1. ISSN: 0360-0300. DOI: 10.1145/3041957.

Lautenbacher, F., Eisenbarth, T., and Bauer, B. (2009). "Process model adaptation using semantic technologies." In: *2009 13th Enterprise Distributed Object Computing Conference Workshops, EDOCW*, pp. 301–309. DOI: 10.1109/EDOCW.2009.5331985.

Le Clair, C. (2013). *Make Business Agility A Key Corporate Attribute – It Could Be What Saves You*. URL: http://blogs.forrester.com/craig_le_clair/13-09-09-make_business_agility_a_key_corporate_attribute_it_could_be_what_saves_you.

Leemans, S. J. J., Fahland, D., and van der Aalst, W. M. P. (May 2018). "Scalable process discovery and conformance checking." In: *Software & Systems Modeling* 17.2, pp. 599–631. ISSN: 1619-1374. DOI: 10.1007/s10270-016-0545-x.

Lin, S.-Y., Lin, G.-T., Chao, K.-M., and Lo, C.-C. (2012). "A Cost-Effective Planning Graph Approach for Large-Scale Web Service Composition." In: *Mathematical Problems in Engineering* 2012.1, pp. 1–21. ISSN: 1024-123X. DOI: 10.1155/2012/783476.

Linden, I., Derbali, M., Schwanen, G., Jacquet, J.-M., Ramdoyal, R., and Ponsard, C. (2014). "Supporting Business Process Exception Management by Dynamically Building Processes Using the BEM Framework." In: *Decision Support Systems III - Impact of Decision Support Systems for Global Environments*. Ed. by F. Dargam, J. E. Hernández,

P. Zaraté, S. Liu, R. Ribeiro, B. Delibašić, and J. Papathanasiou. Vol. 184. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, pp. 67–78. ISBN: 978-3-319-11363-0. DOI: 10.1007/978-3-319-11364-7_7.

Marrella, A. (June 2019). "Automated Planning for Business Process Management." In: *Journal on Data Semantics* 8.2, pp. 79–98. ISSN: 1861-2040. DOI: 10.1007/s13740-018-0096-0.

Marrella, A. and Mecella, M. (2011a). "Continuous Planning for Solving Business Process Adaptivity." In: *Enterprise, Business-Process and Information Systems Modeling*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, T. Halpin, S. Nurcan, J. Krogstie, P. Soffer, E. Proper, R. Schmidt, and I. Bider. Vol. 81. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 118–132. ISBN: 978-3-642-21758-6. DOI: 10.1007/978-3-642-21759-3_9.

Marrella, A., Mecella, M., and Russo, A. (2011b). "Featuring Automatic Adaptivity through Workflow Enactment and Planning." In: *Proceedings of the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Ed. by D. Georgakopoulos and J. Joshi. DOI: 10.4108/icst.collaboratecom.2011.247096.

Marrella, A., Mecella, M., and Sardina, S. (2017). "Intelligent Process Adaptation in the SmartPM System." In: *ACM Transactions on Intelligent Systems and Technology* 8.2, pp. 1–43. ISSN: 21576904. DOI: 10.1145/2948071.

Marrella, A., Russo, A., and Mecella, M. (2012). "Planlets: Automatically Recovering Dynamic Processes in YAWL." In: *On the Move to Meaningful Internet Systems: OTM 2012*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, H. Panetto, T. Dillon, S. Rinderle-Ma, P. Dadam, X. Zhou, S. Pearson, A. Ferscha, S. Bergamaschi, and I. F. Cruz. Vol. 7565. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 268–286. ISBN: 978-3-642-33605-8. DOI: 10.1007/978-3-642-33606-5_17.

Martin, J. (1983). *Managing the data-base environment*. Englewood Cliffs, NJ: Prentice-Hall. ISBN: 9780135505823.

Masellis, R. D., Di Francescomarino, C., Ghidini, C., Montali, M., and Tessaris, S. (Feb. 2017). "Add Data into Business Process Verification: Bridging the Gap between Theory and Practice." In: *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. Ed. by S. Singh and S. Markovitch. AAAI'17. San Francisco, California, USA: AAAI Press, pp. 1091–1099. DOI: 10.5555/3298239.3298400.

McElheran, K. (2015). "Do Market Leaders Lead in Business Process Innovation? The Case(s) of E-business Adoption." In: *Management Science* 61.6, pp. 1197–1216. ISSN: 0025-1909. DOI: 10.1287/mnsc.2014.2020.

Mei-hong, S., Shou-shan, J., Yong-gang, G., Liang, C., and Kai-duan, C. (2012). "A Method of Adaptive Process Mining Based on Time-Varying Sliding Window and Relation of Adjacent Event Dependency." In: *Intelligent System Design and Engineering Application (ISDEA), 2012 Second International Conference on*, pp. 24–31.

Mejri, A., Ayachi-Ghannouchi, S., and Martinho, R. (2020/02/08 2018). "A quantitative approach for measuring the degree of flexibility of business process models." In: *Business Process Management Journal (BPMJ)* 24.4, pp. 1023–1049. DOI: 10.1108/BPMJ-03-2017-0058.

Mendling, J., Verbeek, H. M. W., van Dongen, B. F., van der Aalst, W. M. P., and Neumann, G. (2008). "Detection and prediction of errors in EPCs of the SAP reference model." In: *Data & Knowledge Engineering* 64.1, pp. 312–329. DOI: 10.1016/j.datak.2007.06.019.

Meyer, H. and Weske, M. (2006). "Automated service composition using heuristic search." In: *Business Process Management*, pp. 81–96.

Montarnal, A., Mu, W., Benaben, F., Lamothe, J., Lauras, M., and Salatge, N. (2018). "Automated deduction of cross-organizational collaborative business processes." In: *Information Sciences* 453, pp. 30–49. ISSN: 0020-0255. DOI: 10.1016/j.ins.2018.03.041.

Nebel, B. and Koehler, J. (1995). "Plan reuse versus plan generation: A theoretical and empirical analysis." In: *Artificial Intelligence* 76.1-2, pp. 427–454. ISSN: 0004-3702. DOI: 10.1016/0004-3702(94)00082-C.

Nunes, V. T., Santoro, F. M., Werner, C. M. L., and G. Ralha, C. (2018). "Real-Time Process Adaptation: A Context-Aware Replanning Approach." In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.1, pp. 99–118. ISSN: 2168-2216. DOI: 10.1109/TSMC.2016.2591538.

Pesic, M., Schonenberg, M. H., Sidorova, N., and van der Aalst, Wil M. P. (2007). "Constraint-Based Workflow Models: Change Made Easy." In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, and Z. Tari. Vol. 4803. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 77–94. ISBN: 978-3-540-76846-3. DOI: 10.1007/978-3-540-76848-7_7.

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

Prat, N., Comyn-Wattiau, I., and Akoka, J. (2015). "A Taxonomy of Evaluation Methods for Information Systems Artifacts." In: *Journal of Management Information Systems* 32.3, pp. 229–267. DOI: 10.1080/07421222.2015.1099390.

Rao, J. and Su, X. (2005). "A Survey of Automated Web Service Composition Methods." In: *Semantic Web Services and Web Process Composition*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Cardoso, and A. Sheth. Vol. 3387. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 43–54. ISBN: 978-3-540-24328-1. DOI: 10.1007/978-3-540-30581-1_5.

Regev, G., Bider, I., and Wegmann, A. (2007). "Defining business process flexibility with the help of invariants." In: *Software Process: Improvement and Practice* 12.1, pp. 65–79.

Reichert, M. and Dadam, P. (1997). "A framework for dynamic changes in workflow management systems." In: *Database and Expert Systems Applications, 1997. Proceedings., Eighth International Workshop on*, pp. 42–48.

Reichert, M. and Dadam, P. (1998). "Adeptflex—Supporting Dynamic Changes of Workflows Without Losing Control." In: *Journal of Intelligent Information Systems* 10.2, pp. 93–129. ISSN: 09259902. DOI: 10.1023/A:1008604709862.

Reichert, M. U. and Weber, B. (2012). *Enabling flexibility in process-aware information systems: challenges, methods, technologies*. Springer. ISBN: 3642304095.

Reisert, C., Zelt, S., and Wacker, J. (2018). "How to Move from Paper to Impact in Business Process Management: The Journey of SAP." In: *Business Process Management Cases: Digital Innovation and Business Transformation in Practice*. Ed. by J. vom Brocke and J. Mendling. Cham: Springer International Publishing, pp. 21–36. ISBN: 978-3-319-58307-5. DOI: 10.1007/978-3-319-58307-5_2.

Rinderle, S., Reichert, M., and Dadam, P. (2004). "Correctness criteria for dynamic changes in workflow systems—-a survey." In: *Data & Knowledge Engineering* 50.1, pp. 9–34. DOI: 10.1016/j.datak.2004.01.002.

Rosemann, M., Recker, J. C., and Flender, C. (2010). "Designing context-aware business processes." In: *Systems Analysis and Design: People, Processes, and Projects. Armonk, NY: ME Sharpe, Inc*, pp. 53–74.

Rosemann, M. and vom Brocke, J. (2015). "The Six Core Elements of Business Process Management." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Heidelberg Dordrecht London New York: Springer, pp. 107–122.

Roy, S., Sajeev, A. S. M., Bihary, S., and Ranjan, A. (2014). "An Empirical Study of Error Patterns in Industrial Business Process Models." In: *IEEE Transactions on Services Computing* 7.2, pp. 140–153. ISSN: 1939-1374. DOI: 10.1109/TSC.2013.10.

Scala, E., Micalizio, R., and Torasso, P. (2015). "Robust plan execution via reconfiguration and replanning." In: *AI Communications* 28.3, pp. 479–509. ISSN: 18758452. DOI: 10.3233/AIC-140629.

Song, W., Jacobsen, H., Ye, C., and Ma, X. (Sept. 2016). "Process Discovery from Dependence-Complete Event Logs." In: *IEEE Transactions on Services Computing* 9.5, pp. 714–727. ISSN: 2372-0204. DOI: 10.1109/TSC.2015.2426181.

Tax, N., Verenich, I., La Rosa, M., and Dumas, M. (2017). "Predictive Business Process Monitoring with LSTM Neural Networks." In: *Proceedings of the 29th International Conference on Advanced Information Systems Engineering (CAiSE)*. Ed. by E. Dubois and K. Pohl. Vol. 10253. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 477–492. ISBN: 978-3-319-59536-8. DOI: 10.1007/978-3-319-59536-8_30.

Van der Aalst, W. M. P. (2013). "Business process management: A comprehensive survey." In: *ISRN Software Engineering* 2013.

Van der Aalst, W. M. P. (2015). "Extracting Event Data from Databases to Unleash Process Mining." In: *BPM - Driving Innovation in a Digital World*. Ed. by J. vom Brocke and T. Schmiedel. Management for Professionals. Cham: Springer International Publishing, pp. 105–128. ISBN: 978-3-319-14429-0. DOI: 10.1007/978-3-319-14430-6_8.

Van der Aalst, W. M. P. and Jablonski, S. (2000b). "Dealing with workflow change: identification of issues and solutions." In: *International Journal of Computer Systems Science & Engineering* 15.5, pp. 267–276.

Van der Aalst, W. M. P., Pesic, M., and Schonenberg, H. (2009). "Declarative workflows: Balancing between flexibility and support." In: *Computer Science - Research and Development* 23.2, pp. 99–113.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Van der Aalst, W. M. P. and Verbeek, H. M. (2014). "Process discovery and conformance checking using passages." In: *Fundamenta Informaticae* 131.1, pp. 103–138.

Van der Krogt, R., Bos, A., and Witteveen, C. (2002). "Replanning in a Resource-Based Framework." In: *Multi-Agent Systems and Applications II*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, V. Mařík, O. Štěpánková, H. Krautwurmová, and M. Luck. Vol. 2322. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 148–158. ISBN: 978-3-540-43377-4. DOI: 10.1007/3-540-45982-0_7.

Van der Krogt, R. and de Weerdt, M. (2005). "Plan Repair as an Extension of Planning." In: *ICAPS 2005: Proceedings of the 15th International Conference on Automated Planning and Scheduling, Monterey, California, USA, 5-10 June 2005*, pp. 161–170.

Van Beest, N., Kaldeli, E., Bulanov, P., Wortmann, J. C., and Lazovik, A. (2014). "Automated runtime repair of business processes." In: *Information Systems* 39, pp. 45–79. ISSN: 0306-4379. DOI: 10.1016/j.is.2013.07.003.

Van Dongen, B. F., Alves de Medeiros, A. K., and Wen, L. (2009). "Process Mining: Overview and Outlook of Petri Net Discovery Algorithms." In: *Transactions on Petri Nets and Other Models of Concurrency II*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, K. Jensen, and W. M. P. van der Aalst. Vol. 5460. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 225–242. ISBN: 978-3-642-00898-6. DOI: 10.1007/978-3-642-00899-3_13.

Van Gorp, P. and Dijkman, R. M. (2013). "A visual token-based formalization of BPMN 2.0 based on in-place transformations." In: *Information and Software Technology* 55.2. Special Section: Component-Based Software Engineering (CBSE), 2011, pp. 365–394. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2012.08.014.

Verbeek, H. M. W. and van der Aalst, W. M. P. (2005). "Analyzing BPEL processes using Petri nets." In: *Proceedings of the Second International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management*, pp. 59–78.

Vom Brocke, J. (2009). "Design Principles for Reference Modelling." In: *Innovations in Information Systems Modeling*. Ed. by T. Halpin, J. Krogstie, and E. Proper. IGI Global, pp. 269–296. ISBN: 9781605662787. DOI: 10.4018/978-1-60566-278-7.ch014.

Vom Brocke, J. and Mendling, J., eds. (2018). *Business Process Management Cases: Digital Innovation and Business Transformation in Practice*. Management for Professionals. Cham: Springer International Publishing. ISBN: 978-3-319-58306-8. DOI: 10.1007/978-3-319-58307-5.

Weber, B., Reichert, M., and Rinderle-Ma, S. (2008a). "Change patterns and change support features–enhancing flexibility in process-aware information systems." In: *Data & Knowledge Engineering* 66.3, pp. 438–466.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Wynn, M. T., Verbeek, H. M. W., van der Aalst, W. M. P., ter Hofstede, A. H. M., and Edmond, D. (2009). "Business process verification-finally a reality!" In: *Business Process Management Journal* 15.1, pp. 74–92.

Zheng, X. and Yan, Y. (2008). "An Efficient Syntactic Web Service Composition Algorithm Based on the Planning Graph Model." In: *2008 IEEE International Conference on Web Services (ICWS)*, pp. 691–699. DOI: 10.1109/ICWS.2008.134.

4

Evaluation of Automated Planning with respect to the Efficiency of Process Modelers

This Chapter comprises one paper that addresses FT1 and in particular RQ3 of the dissertation at hand. Paper 5 aims at answering the question whether Automated Planning of Process Models is increasing the task performance of process modelers and hence whether it offers an economical advantage in contrast to a "traditional" well-known process modeling approach. Thereby, it is not sufficient to only consider the time that is required for a modeling task but also the quality of the resulting process models should be taken into account. Paper 5 strives at evaluating this by means of a laboratory experiment.

4.1 Paper 5: The Influence of Automated Planning on the Task Performance of Process Modelers

Status	Published	Full Citation
Accepted	12/2019	Schön, D. (2019), The Influence of Automated Planning on the Task Performance of Process Modelers, Proceedings of the 40th International Conference on Information Systems (ICIS 2019), December 15-18, Munich, Germany.

Abstract

Constructing and adapting process models is highly relevant in today's business world, but time-consuming and error-prone. Several approaches address these issues by manual tasks or making use of automation in the past years. Especially the research field Automated Planning envisions a (semi-)automated construction of process models by using semantic annotations and planning techniques.

We aim at an empirical analysis of the influence of Automated Planning on the task performance of process modelers compared to the task performance when using common process modeling tools. We analyze the invested effort in terms of the required time for modeling tasks and the outcome in terms of the quality of the constructed process models by means of a laboratory experiment.

Our findings indicate that Automated Planning significantly improves the task performance. The quality of constructed process models is increased, and especially for larger process models, the required time for modeling tasks could be decreased.

Keywords: Business Process Management, Automated Planning, Experimental Evaluation, Task Performance

Post publication changes:

- Several formatting changes for consistency reasons
- Changed citation style for consistency reasons
- The numbering of headlines, tables, figures and definitions was changed for consistency reasons
- Consistent naming for variables $corr_{synt}$, $corr_{sem}$, $comp_{sem}$
- Consistent capitalization of Section references

4.1.1 Introduction

Process models not only allow representing processes but are crucial instruments for decision-makers (Rosemann et al., 2015), support the alignment of companies' IT infrastructure with their business strategy (Branco et al., 2014) and are an essential tool in business reorganization projects (Becker et al., 2010a; Mendling et al., 2010). Formalized, imperative process models are commonly used as a basis for a semi-automated execution of processes (Ding et al., 2015; Paik et al., 2014; Wang et al., 2014) or their verification (e.g., Wetzstein et al., 2007). Additionally, a suitable representation of processes using process models is a crucial success factor for the development of service-oriented information systems (Aguilar-Savén, 2004; Mendling et al., 2012a). However, today's process modeling projects are facing several issues in practice (i.e., in a business context). Process modeling and improving processes is time-consuming (Heinrich et al., 2015b; Hornung et al., 2007; van der Aalst et al., 2000b) as it is not supported by algorithms mainly. A survey conducted by Becker et al. (2010b) among 60 banks underlines this: "over two thirds [...] have a negative effort-utility-ratio concerning their process modeling initiatives". In a cooperative project, we analyzed over 600 core business processes as well as 1,500 support processes of a European bank. Several IT and business executives of the bank stated that today's process (re)design projects are more cost-intensive and time-consuming than ten years ago, due to higher complexity. This is particularly relevant as the know-how of representing business processes in a formal and syntactically as well as semantically correct way is completely missing (Becker et al., 2015) or in the responsibility of a few specialists in many organizations.

Hence, when talking about an effort-utility-ratio of business process modeling, assessing the required time (i.e., personnel costs) is not sufficient. With missing know-how, quality issues of process models are more likely (Becker et al., 2006; Kusiak et al., 1994; Mendling et al., 2007a). Thus, the "value of business process modeling", seen as one of the leading challenges and issues for practitioners in the field of Business Process Management (BPM) by Indulska et al. (2009), could be decreased due to errors during process modeling. Empirical studies of Roy et al. (2014) and Fahland et al. (2011) show that up to 92.9 % of process models are erroneous in business contexts. Besides semantic errors like missing actions, in particular, syntactic errors (e.g., ambiguous gateways) are contained in these process models. Even though such errors do not render the models worthless, they make it very difficult to use them to apply, for instance, approaches for automated verification (Weber et al., 2008b) and execution (Khan et al., 2010; Weber, 2007) of service-oriented systems. As said, usually few specialists are responsible for process modeling by translating the input of process participants (e.g., staff from

specialist departments) into formalized process models. If this translation fails, process modeling projects fail as well (Rosemann, 2006).

On the other hand, employees' implicit knowledge is essential when constructing process models (cf. Seethamraju et al., 2009). Especially, when business processes span across different departments or companies, this is even more relevant as participants tend to have isolated views only on parts of processes within their organizational unit (cf. Gordijn et al., 2000). For instance, we were facing two significant challenges in a large-scale business process reengineering project of a German insurance in which the authors were involved: the retrieval of complete and correct information of how processes are executed in reality and a precise definition of conditions that regulate, when and by whom parts of processes may be executed. Hence, it is promising to deeply involve employees with strong explicit knowledge of how parts of processes are executed in process modeling projects (Rosemann, 2006).

Thus, enabling process participants (novices regarding process modeling) to construct process models and allowing them to construct process models (cf. Scholtz et al., 2013) on their own is promising. It seems favorable to enable employees capturing their specific work tasks from an insider's perspective (Fleischmann et al., 2013) instead of specialists constructing process models with an outsider's perspective. Guidance for constructing formalized, imperative process models, using algorithms, especially for novices (Recker et al., 2012), may help to achieve this objective. The research strand of Automated Planning of Process Models (we refer to *Automated Planning* in the following) envisions to construct process models (semi-)automatically (Heinrich et al., 2012; Heinrich et al., 2015b; Henneberger et al., 2008). A combination of semantic annotations, as envisioned in Semantic Business Process Management (Becker et al., 2015; Betz et al., 2006; Brockmans et al., 2006; Hepp et al., 2007; Hepp et al., 2005; Thomas et al., 2009), and existing planning techniques (Bertoli et al., 2006; Hoffmann et al., 2005) is used. Particularly, a planner generates a feasible process model based on a set of semantically annotated actions, an initial state that represents the starting point of a process and goal states representing the desired outcome of the process (see Section 4.1.3). However, it is still to evaluate whether novices are able to specify such information and whether the resulting process models are less erroneous. Even though a higher degree of automation is envisioned, for instance, the efforts required for the (initial) annotation of actions are expected to be higher, compared to common process modeling approaches (cf., e.g., Heinrich et al., 2015b; Krause et al., 2013). We, therefore, follow previous works and evaluate the "task performance of users" of Automated Planning, one of the most common evaluation criteria, according to Prat et al. (2015).

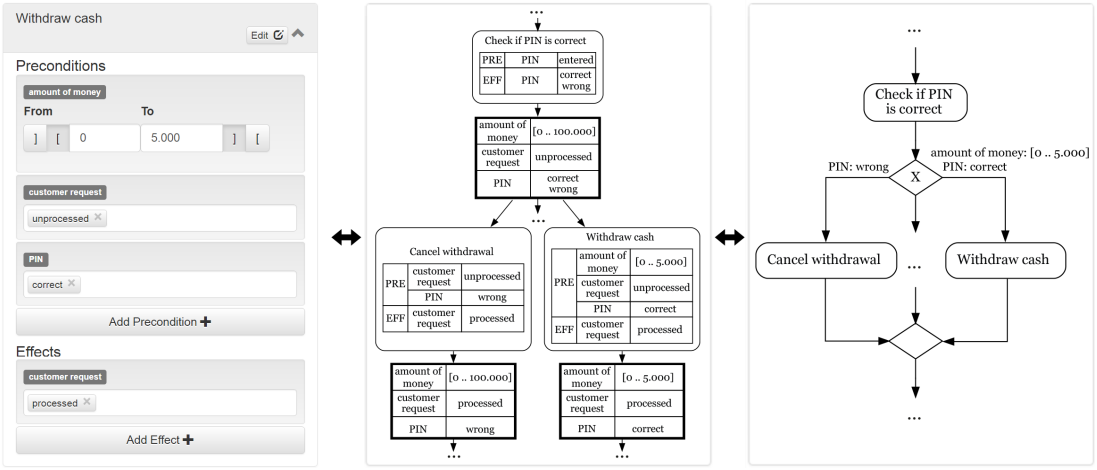

Figure 4.1: Basic Steps of the Automated Planning Approach
(left: annotated action in web-based tool; center: excerpt of Action State Graph; right: excerpt of process model)

4.1.2 Research Objective & Context

We aim at evaluating whether Automated Planning influences the task performance of process modelers. Therefore, we conduct a laboratory experiment. Jedlitschka et al. (2008) propose a structured approach for conducting and reporting empirical research in the field of Information Systems, which strongly influences the structure and contents of this paper. In the next Section, we briefly introduce the incorporated Automated Planning approach and outline the differences between common process modeling (particularly using ARIS Express) and Automated Planning. Thereafter, we introduce the plan of our experiment. In the main part of this paper, we describe the results of the experiment, interpret them, and discuss the validity of our findings. We close with limitations and an outlook on further research.

The study consists of one controlled experiment conducted with 70 students, who enrolled in the course "Programming in Practice" (not-mandatory part of the bachelor's in management of Information Systems curricula) at a German university. Students can be considered as novices in process modeling (cf. Recker et al., 2012) but, according to Kitchenham et al. (2002), using students "is not a major issue as long as you are interested in evaluating the use of a technique by novice[s] or nonexpert[s]", the primary goal of this study.

4.1.3 Related Work and Technology under Investigation

Several research fields along the BPM Lifecycle (Scheer et al., 2010; van der Aalst, 2013; zur Muehlen et al., 2004) support modelers and business analysts via automatic techniques as "BPM should make opportune use of technology" (vom Brocke et al., 2014). It seems promising to provide guidance in constructing formalized, imperative process models, especially for novices (Recker et al., 2012). Plenty of these existing approaches rely on semantic annotations and integrate semantic technologies in process modeling and analysis (cf., e.g., Fellmann et al., 2015a). In the process analysis phase, for instance, process mining, as well as process model verification and validation, assist analysts (e.g., Wetzstein et al., 2007). Automated (web) service selection and composition can be seen as part of the phases process implementation and process execution (Ding et al., 2015; Paik et al., 2014; Wang et al., 2014). In the process modeling phase the research strand of Automated Process Planning envisions the construction of process models by means of algorithms (cf., e.g., Heinrich et al., 2009; Heinrich et al., 2012; Henneberger et al., 2008; Hoffmann et al., 2012; Lautenbacher et al., 2009) and aims at increasing the "flexibility by definition" (van der Aalst, 2013) of process models and to reduce the manual efforts when constructing process models (Heinrich et al., 2012; Heinrich et al., 2015b; Henneberger et al., 2008). In the remainder, we describe Automated Planning, the technology under investigation, and present differences between process modeling, using Automated Planning and process modeling using common process modeling tools (i.e., alternative technologies).

Automated Planning constructs imperative process models consisting of actions and control-flow patterns (e.g., exclusive choice and parallelization) based on a formalized input of annotated actions, an initial state and one to many goal states. The annotation of actions consists of preconditions (criteria to conduct the action) and effects (how the action affects the state of the process) which are represented by variables (e.g., "amount of money"; cf. leftmost in Figure 4.1). Each variable depicts one specific aspect of the represented process. The approach abstracts the construction of (graphical) process models from ordering actions and control-flow patterns to specifying contentual information about the process and the actions the process comprises. This is the (evaluated) manual task, a modeler has to perform when constructing process models using Automated Planning. Based on this input, the annotated actions, an initial state as well as the goal states (sets of variables, too; input of the approach), a *planner* (i.e., a set of algorithms) automatically constructs feasible process models (output of the approach) in three automated phases A1) to A3):

A1) Retrieval of dependencies between actions: When the modeler finishes entering the information as mentioned above, he/she initiates the actual automated plan-

ning. First, the planner retrieves dependencies between actions. It analyzes the actions pairwise regarding their preconditions and effects. Thereby, it retrieves whether, for instance, preconditions or effects of an action are missing to construct the desired process model successfully. It further avoids repeated (redundant) analyses in subsequent steps and hence improves the overall performance of the planner. In this step, algorithms identify all actions that possibly "contribute" to reaching the goal state(s) of the desired process model.

A2) Planning feasible sequences of actions: A depth-first forward search determines predecessor-successor-relationships between actions. Starting with the initial state, A2i) actions that are applicable in this state are retrieved (i.e., are preconditions fulfilled?). Then, A2ii) the following state for each applicable action is constructed by applying the preconditions and effects of the action on the preceding state (the initial state in the first iteration). After that, the subphases A2i) and A2ii) are performed recursively in each created following state. The forward search stops if all possible sequences of actions leading from the initial state to the goal state(s) are found and a complete Action State Graph is constructed (cf. center of Figure 4.1).

A3) Construction of feasible process models with control-flow patterns: To construct feasible process models with control-flow patterns, determining sequences of actions is not sufficient. Therefore the planner constructs well-known control-flow patterns (Russell et al., 2006a; Russell et al., 2016; van der Aalst et al., 2003) in the last step (cf., e.g., Heinrich et al., 2009; Heinrich et al., 2015a). The approach relies on an abstract representation of process models. Thus, the construction of control-flow patterns is not restricted to a particular process modeling language.

The tool, used for this experiment (i.e., a scientific prototype), is able to generate UML Activity Diagrams (cf. rightmost area of Figure 4.1; *OMG Unified Modeling Language TM (OMG UML): Version 2.5* 2015) but could be extended to generate other imperative process modeling languages (e.g., BPMN, eEPC, WS-BPEL). The algorithms are able to construct the most common control-flow patterns (e.g., exclusive choices, simple merges, parallelizations, and synchronizations) currently. It allows to define actions and annotations via a web-based user interface (see `http://www-sempa.uni-regensburg.de` for a public version of the prototype), Modelers define the preconditions and effects of actions in a formalized manner. This formal specification is the task under investigation. In particular, the tool allows defining one action after the other in a list-oriented way, regardless of the control-flow of the process which is constructed in an automated manner. Modelers define new actions by entering a unique name for the action and specifying its preconditions and effects by selecting variables as well as their possible values from lists. They select one to possibly many variables from a list and choose the according values in a second step. Figure 4.2 exemplarily illustrates the task of adding precon-

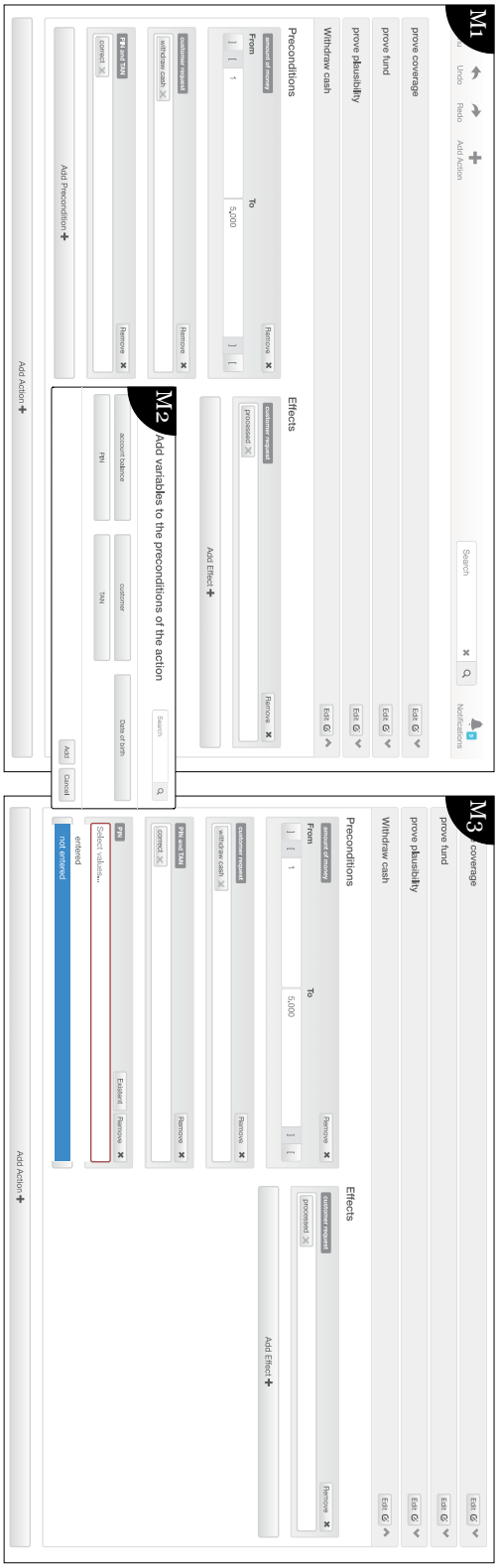

Figure 4.2: User Interface of the Automated Planning Tool when adding a Precondition

ditions to the action "Withdraw cash" in the web-frontend. To conduct the action, the variable "amount of money" has to be between 1 and 5.000, "customer request" has to be "withdraw cash", "PIN and TAN" has to be "correct". The effect of the action is that "customer request" is set to "processed".

Additionally, the modeler adds the precondition that the variable "PIN" has to be "entered". To do this, in step M1 (see Figure 4.2), he/she clicks the "Add Precondition" button. After that, the tool presents possible variables (step M2). After choosing the according (here: "PIN"), the tool presents possible values for the variable (step M3) from which the modeler chooses. In addition to the annotated actions, the modeler has to define an initial state (representing the starting point of the process) and one or more goal states (representing the desired outcome of the process) similarly. In order to be able to determine possible sequences of actions manually, modelers using a common process modeling tool need to define (at least implicitly) preconditions and effects of actions as well as initial and goal states, too. However, for enabling the Automated Planning approach (phases A1) to A3)), they have to be formally and explicitly specified.

When constructing process models using a common process modeling tool, process modelers, on the other hand, need to arrange the actions of a process model in the appropriate order. Additionally, they have to construct control-flow patterns where necessary to create formalized process models in a particular process modeling language (e.g., eEPC, BPMN). From a modeler's perspective, common process modeling tools support constructing business process models "sequence-oriented". In particular, a modeler usually starts to define a start node of the process model. Then, the modeling tool presents a set of possible "next" node types (i.e., e.g., actions, control-flow patterns) to the modeler. He/she selects one of these and, in case of selecting an action, the modeler enters its name. After that, the modeling tool again presents a set of possible following node types. Hence, to be able to create an entire, feasible process model, a modeler has to construct an (at least vague) mental model of the entire process. Otherwise, he/she would not be able to incorporate such a "sequence-oriented" modeling approach.

To sum up, we presume the more formal specification of actions taking more time when using Automated Planning. On the other hand, modelers do not have to identify contributing actions as they are identified automatically during phase A1). Thereafter, the Automated Planning approach retrieves the feasible ordering of actions (phase A2)) as well as necessary control-flow patterns (phase A3)) automatically, based on the given set of annotated actions as well as initial and goal states, the modeler provides.

4.1.4 Experiment Planning

In this Section, we describe the plan of the experiment that is used to perform it and to analyze the results.

4.1.4.1 Goals

Precisely, we state the overall experimental goal:

> Analyze an Automated Planning approach for (semi-) automatically constructing process models and a commonly used process modeling tool for the purpose of comparing them with respect to the task performance of process modelers from the point of view of the researcher in the context of a Management Information Systems lecture at a German university.

Performance is commonly defined as obtained outcome divided by invested resources. In the context of process modeling, this could be the quality (in terms of completeness and correctness) of the constructed process models (obtained outcome) divided by the time, needed for the modeling task (invested resources).

This leads us to formulate a different goal for each facet.

> Goal G_A: Analyze an Automated Planning approach ... and a commonly used process modeling tool for the purpose of comparing them with respect to the obtained outcome of modeling tasks ...
>
> Goal G_B: Analyze an Automated Planning approach ... and a commonly used process modeling tool for the purpose of comparing them with respect to the required time of modeling tasks ...

Overhage et al. (2012) propose the 3QM-Framework to determine the quality of process models systematically. They state that process models – amongst others – shall comply with the syntactical rules as well as semantically comply with the according real-world excerpt (in terms of completeness and correctness). In line with their approach, we divide Goal A into three sub-facets, each of which is covering one particular aspect of the obtained outcome of the modeling tasks:

> Goal G_{A1}: ... with respect to the syntactic correctness of constructed process models ...
> Goal G_{A2}: ... with respect to the semantic correctness of constructed process models ...
> Goal G_{A3}: ... with respect to the semantic completeness of constructed process models ...

	P1 Mobile contract	*P2* Stationary internet contract	*P3* Cash withdrawal/ deposit	*P4* Local electronics tore
Total num. of action / control-flow pattern occurrences	11 / 10	35 / 26	12 / 10	35 / 26
Total num. of variable occurrences	27	126	35	132

Table 4.1: Key Figures of the Process Models, considered in the Experiment

4.1.4.2 Participants

70 bachelor students in the field of Management of Information Systems participated in the experiment. It was part of courses, and the covered topic (i.e., the automated planning approach) was relevant for the exam. The experiment was part of the not mandatory practice lessons allowing the students to exercise for the final exams. Participating in the experiment was voluntary. Informed consent has been obtained by informing the students in advance about the experiment and stating that participation was voluntary. In advance of the actual experiment, coworkers of the authors (research assistants) participated in the pretest. We assigned participants randomly to one group whereas the groups differed in the selection of the process modeling tasks, their order (e.g., first the adaptation of the small process model, then modeling the large process model from scratch) and the tool the group had to use.

4.1.4.3 Experimental Material and Tasks

To analyze the influence of Automated Planning on the task performance of process modelers, we created textual descriptions of four processes. We aim at simulating real-world modeling projects, where system analysts and domain experts talk about to-be modeled processes in the first step (cf., e.g., Frederiks et al., 2006; Hoppenbrouwers et al., 2005). Hence, we structured the textual descriptions in the form of interview transcripts. The processes are generally admitted, simplified versions of *enrollment of a mobile contract* (*P1*), *enrollment of a stationary internet contract* (*P2*), *cash withdrawal and deposit* (*P3*) as well as *shopping in a local electronics store* (*P4*). These processes were derived from unstructured interviews with professionals in the according industries and simplified for the purpose of this experiment. We rely on these processes as each participant presumably has at least once experienced them and influences by *domain knowledge* could largely be alleviated. The process models differ in their size, i.e., the total number of

action and control-flow pattern as well as variable occurrences. Table 4.1 summarizes these measures of the four "target process models", considered in our experiment. *P1* and *P2* have to be modeled from scratch. *P3* and *P4* have to be adapted significantly. In *P3*, security checks become obsolete in case of cash deposit and an account balance enquiry should be considered additionally in case of withdrawal. *P4* has to be extended: a customer has to be informed that ordered products are ready for collection and the subprocess of a customer collecting goods needs to be complemented. Further, options to use self-service terminals have to be included.

4.1.4.4 Hypotheses, Parameters, and Variables

As seen previously, constructing process models differs in several aspects when comparing the Automated Planning of process models and a common process modeling approach. Hence, we expect differences in the task performance of process modelers. We, therefore, chose the independent variable of our research model *Use of Automated Planning,* and the dependent variable *task performance of the process modeler*.

Therefore, we formulate the null hypothesis for our experiment

H_0: The use of Automated Planning does not *influence* the task performance of process modelers.

We divide the participants into two experimental groups concerning the treatment, the *use of Automated Planning*. One group uses the Automated Planning tool (i.e., a scientific prototype), the other uses ARIS Express, a common process modeling tool. To comprehensively investigate the performance of the process modeler, we rely on two facets of the performance. As mentioned in Section *Goals*, we aim at comparing two technologies with respect to the obtained outcomes of a task and the invested time for the task. We rely on the quality of the resulting process models as a measure for the obtained outcomes. In a literature review on quality issues of process models, conducted by Moreno-Montes de Oca et al. (2015), the semantic and syntactic quality belong to the most frequently referred. Becker et al. (2000) describe "syntactic correctness" as the consistency and completeness against the underlying metamodel of the according process model and semantic correctness as the consistency of a model with the real world. Leopold et al. (2016) state that, in about 77% of 585 process models (denoted in BPMN 2.0) data objects that are not linked to an existing glossary are used. Hence, the semantic quality, in particular, the consistency between different models representing the same process (Becker et al., 2000; Zamperoni et al., 1993), is threatened. Hence, we aim at evaluating how far the resulting process models are consistent with a given desired "target process model".

Previous works mathematically prove that the Automated Planning approach con-

structs correct, minimal and complete process models (cf., e.g., Bertoli et al., 2006; Heinrich et al., 2015b). Thus, we especially expect increased syntactic correctness of the resulting process models when using Automated Planning. Though common process modeling tools support modelers to avoid syntactical errors by tooltips and hints, they do not prevent it. Further, Automated Planning enables modelers to specify one action after the other. This is due to the fact that in contrast to when using a common process modeling tool, a modeler does not have to cope with the whole process model at once. He/she has not to cope with dependencies between actions that may have consequences on other parts of a process model. Therefore, we assume that the semantic correctness could potentially be influenced as well. As previously mentioned, Automated Planning (in particular phase A1) of the approach) identifies actions that are necessary and relevant for a process model. Further, in phase A3), the approach identifies where to construct which control-flow pattern. These two capabilities of the approach let us assume that especially the semantic completeness of process models (i.e., all necessary actions and control-flow patterns are contained in the process model) could be increased.

To assess the influence of Automated Planning on the quality of process models, we aim at using objective metrics for assessing the quality of the resulting process models in our research. Overhage et al. (2012) propose the 3QM-Framework – a refinement of the SEQUAL framework, proposed by Lindland et al. (1994) and adapted for process modeling by Krogstie et al. (2006) – that provides specific quality metrics and measurements that allow "to systematically determine the quality of process models". Here, we focus on ❶ the syntactic correctness $corr_{\text{synt}}$, ❷ the semantic correctness $corr_{\text{sem}}$ and ❸ the semantic completeness $comp_{\text{sem}}$ of the process models. Overhage et al. (2012) propose to use a degree of correctly respectively consistently modeled elements (i.e., belief states, actions, control-flow patterns and edges). We slightly adapt the metrics to the context of our experimental setting and use the metrics, as seen in Table 4.2. Krogstie et al. (2006) propose several other "levels of quality" such as "pragmatic quality" that refers to the interpretation of process model. As we aim at an objective evaluation, we particularly focus on the semantic and syntactic quality for this experiment. To support the statistical validity of our findings(cf. Shadish et al., 2002), we operationalize the quality of resulting process models by the following first hypotheses:

H_A: The use of Automated Planning *increases* the…

H_{A1}: …syntactic correctness of constructed process models.

H_{A2}: …semantic correctness of constructed process models.

H_{A3}: …semantic completeness of constructed process models.

Besides this, the invested time to construct the according process model is the second major factor influencing the task performance of a process modeler. According to Hor-

Name of the variable	Type of the variable	Class	Entity	Type of attribute	Scale type	Unit	Range
use of Automated Planning (treatment)	independ.	method	used modeling approach	internal	nominal	boolean	"true" resp. "false"
syntactic correctness ($corr_{\text{syntactic}}$)	dependent	product	resulting process model	internal	ratio	ratio of elements that are modeled syntactically correct	0 to 1
semantic correctness($corr_{\text{semantic}}$)	dependent	product	resulting process model	internal	ratio	ratio of elements that are modeled semantically correct	0 to 1
semantic completeness($comp_{\text{semantic}}$)	dependent	product	resulting process model	internal	ratio	1-(num. of superfluous elements + num. of missing elements) / num. of req. elements	0 to 1
required time for modeling task	dependent	process	modeling task	internal	absolute	minutes	1 to infinity
type of the modeling task	moder.	process	modeling task	internal	nominal	n/a	"Modeling from scratch" resp. "Adapting an existing process model"
size of process model	moder.	product	desired process model	internal	ordinal	n/a	"small" resp. "large"

Table 4.2: Variables, used in the Experiment

nung et al. (2007), "manual process modeling is a time-consuming task and thus increases the total amount of modeling time." Over two-thirds of 60 banks surveyed complained about a negative effort-utility-ratio of their process modeling projects (cf. Becker et al., 2010b). Krause et al. (2013), as well as Bandara et al. (2006), refer to the "required person days" (resp. "invested person days of effort") as one of the key cost drivers for process redesign projects and process modeling in general. Hence, as the way of constructing process models is essentially different between the compared approaches, it has to be evaluated whether the required time for the overall task changes. When using the Automated Planning approach, modelers need to define the annotation of actions as well as the initial state and goal states of the to be constructed process models.

In particular, he/she has to explicitly formalize this information in contrast to a rather implicit consideration when constructing process models by means of a common process modeling tool. However, algorithms visualize the process model, as well as determine the correct order of actions and control-flow patterns, completely. Thus, we use ❹ the *required time for the modeling task* as a metric for the "invested resources" for process modeling tasks. The required time for the modeling task is derived from log entries (i.e., time between opening and saving the according modeling task) when using the Automated Planning approach. When using ARIS Express, the experimenters recorded the required time manually. We derive the hypothesis, evaluated by means of the following experiment:

H_B: The use of Automated Planning *decreases* the required time for the modeling task.

Besides the independent variable (using a common process modeling tool or an Automated Planning approach) and the dependent variables (❶ to ❹), we discuss the moderating variables in the following. We have to differentiate between "moderating factors" and "modeling related factors" affecting the task performance of process modelers (cf. Bandara et al., 2005; Rosemann et al., 2001; Sedera et al., 2002). Here, moderating factors refer to the process model that should be constructed and its characteristics (e.g., size and type of modeling task) and modeling related factors relate to, among others, *personal factors* of the modeler like his/her experience concerning process modeling. We build on this differentiation and consider *process model factors* and *personal factors* as moderating variables in our experiment.

It is proposed to consider established metrics for *process model factors*, such as the *size of process models* (cf. La Rosa et al., 2011c; Mendling et al., 2008). According to Mendling et al. (2008), especially larger (i.e., 40 actions and more) and complex process models contain more errors. We take this into account by the design of the materials as described in the previous Section. Further, we consider the *type of the modeling task* (i.e., modeling from scratch or adapting existing process models) as moderating variable. We thereby aim at evaluating whether it influences the task performance (especially $corr_{\text{syntactic}}$)

if a given process model is present for the according modeling task. If, for instance, unexperienced process modelers should adapt a given process model, they may align their changes to the existing process model. Without a given process model, they have no possibility to follow an already existing way of visualizing a process and hence may construct process models containing more syntactical errors. Regarding *personal factors* of the process modeler (cf. Recker et al., 2012), others differentiate between *modeling expertise* and *domain knowledge* (cf. Hornung et al., 2008; Moreno-Montes de Oca et al., 2015; Recker et al., 2012; Wang et al., 2007). Modeling expertise describes how familiar someone is regarding the construction of process models. Domain knowledge refers to the subject area of the according process. The participants have been questioned to self-assess their *modeling expertise*. However, all participants could be considered as novices. We renounced to question the domain knowledge but mitigated the influence of this personal factor by relying on generally admitted processes. Table 4.2 summarizes all variables used in our experiment.

4.1.4.5 Experimental Design

We conducted a laboratory experiment to investigate the influence of using an Automated Planning approach on the task performance of process modelers. In detail, we investigate differences in the metrics ❶ to ❹ for the task performance of process modelers between the construction of process models using a common modeling tool as well as an Automated Planning approach. We rely on a 2^k factorial design for our experiment. We aim at evaluating whether the treatment (i.e., using the technology under investigation) influences the task performance of process modelers in different scenarios. Hence, we consider the size of the process models and the type of the modeling task as additional factors. We further use a randomized complete block design as a participant has to solve different tasks in a given, randomized order. The experiment was conducted pursuant to the "Experiment Process" as shown by Wohlin et al. (2012). It was not conducted in an industrial modeling project. It did not focus on a particular modeling language and dealt with the construction of process models – a real problem encountered in BPM projects of every kind.

4.1.4.6 Procedure

All participants were familiar with the common process modeling tool. However, they had no experience in using Automated Planning and the tool at hand. Hence, we explained the Automated Planning approach and tool to the participants a week prior to the actual experiment and asked them to solve modeling tasks as homework to increase comparability. At the beginning of the experiment, we randomly assign each participant

to a numbered desk. When all are seated, we give a brief introduction to the experiment and hand out the *Experimental Material*. After all participants have received the textual process descriptions, they are asked to start constructing the described processes in the given order. Hereby, each participant is assigned to one treatment. Participants at desks with even numbers use a common modeling tool, participants at desks with odd numbers use Automated Planning. The participants do not receive a visualization of the processes that should be modeled from scratch. The process models that should be adapted are prepared in both tools used. When a participant finishes the last task, he/she is asked to leave the room. The experimenters stay in the room for the whole experiment assuring there was no collaboration among the students. After the experiment, the constructed process models were compared to the desired target process models by the experimenters. Thereby, each element of the constructed process models is counted and, based on the counting rules (cf. Table 4.2), the dependent variables are calculated.

4.1.5 Analysis and Discussion

In this Section, we at first present the results of our experiment, discuss them, and highlight implications for research as well as for practice. Thereafter, we discuss possible threats to the validity of our experiment.

4.1.5.1 Descriptive Statistics and Hypothesis Testing

135 process models were constructed successfully. Another five modeling tasks had been canceled (i.e., the participants did not declare the tasks "finished" due to running out of time). The 135 valid samples spread across eight different groups based on the *use of Automated Planning* (Automated Planning vs. common modeling tool), the *process model size* (small vs. larger, cf. Table 4.1), and the *type of the modeling task* (adaptation vs. modeling from scratch). The number of samples varied between 15 and 19 across the groups. As the dependent variables ❶-❸ are not normally distributed, a series of Wilcoxon rank-sum tests have been conducted, to evaluate the hypothesis H_A. As ❹ the required time for the modeling task follows a normal distribution in each class, we validated the results by means of t-tests. The tests have been executed using R version 3.2.3. In the following, we present the descriptive statistics as well as the results of the accordingly used test for each hypothesis. Figure 4.3 shows box plots for all tested hypotheses. Table 4.3 summarizes the descriptive statistics (M: mean; SD: standard deviation) for all hypotheses and the results of the Wilcoxon rank-sum test (W: rank-sum; p: p-value) respectively t-test.

H_{A1}: *The use of Automated Planning increases the syntactic correctness of constructed process models.* There is a highly significant difference in the syntactic correctness $corr_{\text{synt}}$

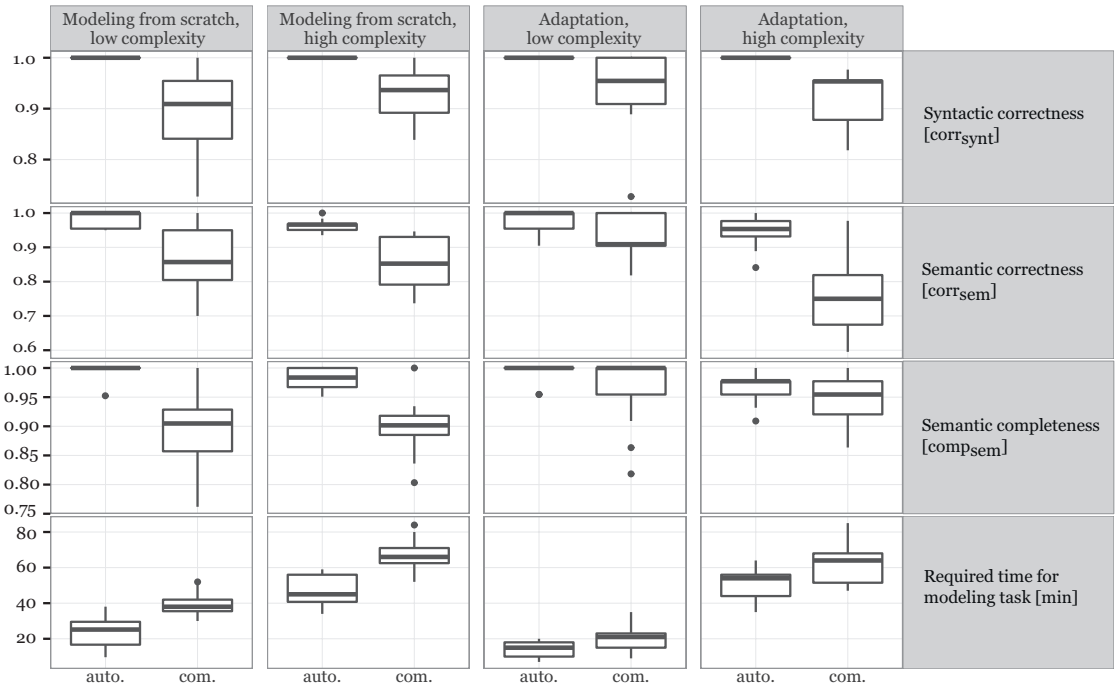

Figure 4.3: Box Plots for all tested Hypotheses

(cf. ❶) of process models constructed using a common process modeling tool ("com.") and automatically planned process models ("auto.") regardless of their size and the type of the modeling task ("Modeling from scratch" resp. "Adapting an existing process model"). These results suggest that when process modelers use Automated Planning, they construct process models with higher syntactic correctness in all cases. We, therefore, accept H_{A1}.

H_{A2}: *The use of Automated Planning increases the semantic correctness of constructed process models.* When modeling from scratch, there is a highly significant difference in the semantic correctness $corr_{sem}$ (cf. ❷) of process models between Automated Planning and using a common process modeling tool regardless of the size of the process models. For the adaptation of existing process models, using Automated Planning creates significantly better results for small process models. When comparing the results for larger process models, the differences are highly significant. These findings (cf. Table 4.3) support that Automated Planning improves the semantic correctness of the resulting process models. Although the semantic correctness of commonly constructed process models is rather high, too, Automated Planning significantly improved it across all cases. Thus, we can accept hypothesis H_{A2}.

H_{A3}: *The use of Automated Planning increases the semantic completeness of constructed pro-*

cess models. The Wilcoxon rank-sum tests show that when adapting process models, Automated Planning does not lead to a significantly higher semantic completeness $comp_{\text{sem}}$ (cf. ❸; Table 4.3). However, Automated Planning creates highly significantly better results when constructing process models from scratch. We are able to accept hypothesis H_{A3} in these two groups. As process models in practice tend to be larger than the small and rather easy process models, considered in our research, the results support that Automated Planning allows process modelers to construct process models with higher semantic completeness.

H_B: *The use of Automated Planning decreases the required time for the modeling task.* The analysis of the boxplots (cf. Figure 4.3) shows that the required time for modeling process models from scratch or adapting them (cf. ❹) is reduced by using Automated Planning. T-tests (cf. Table 4.3) show that using Automated Planning implies a highly significant difference in the required time to adapt or construct process models from scratch regardless of their size. Hypothesis H_B can, therefore, be accepted.

In summary, the results show a significant difference between constructing process models using a common process modeling tool and constructing process models with Automated Planning for all of our hypotheses and thus, $\mathbf{H_0}$ could be rejected.

4.1.5.2 Discussion of the Results

The results indicate that the task performance of process modelers (❶-❹) is positively influenced by using Automated Planning. We assume that process modelers may be able to focus on one action after the other in a sequential manner instead of having the control-flow of the complete, resulting process model in mind by using an Automated Planning approach. This assumption is supported by the findings of our research, particularly by the increase of ❷ the semantic correctness and ❸ the semantic completeness of resulting process models. This is of particular interest as especially the automated identification of issues regarding ❷ and ❸ is hard (cf. Soffer et al., 2012). Our findings support the initial assumption as the semantic quality (❷ and ❸) of small process models is at a rather high level, even when constructing process models with a common process modeling tool. Here, the consideration of the structure of the resulting process models is not that complex, so that process modelers are still able to focus on the semantics of the actions. Further, using an Automated Planning approach assures to construct syntactically correct process models (cf. ❶) by design of the incorporated algorithms. The required time for a modeling task ❹ could be significantly decreased by using Automated Planning. According to the findings of Krause et al. (2013) Automated Planning is especially beneficial for large process models and is not proposed to be used for small process models that are not required to be redesigned frequently due

	Modeling from scratch				Adapting an existing process model			
	Small process model		Larger process model		Small process model		Larger process model	
$\boldsymbol{H}_{A1}$: *The use of Automated Planning increases the syntactic correctness of constructed process models.*								
	com.	auto.	com.	auto.	com.	auto.	com.	auto.
Descriptive statistics [$corr_{synt}$]	M=.897 SD=.088	M=1 SD=0	M=.931 SD=.052	M=1 SD=0	M=.947 SD=.069	M=1 SD=0	M=.920 SD=.053	M=1 SD=0
Res. of Wilc. rank-sum tests	W=332.5 p=0.000 ****		W=216.0 p=0.000 ****		W=229.5 p=0.000 ****		W=255.0 p=0.000 ****	
$\boldsymbol{H}_{A2}$: *The use of Automated Planning increases the semantic correctness of constructed process models.*								
	com.	auto.	com.	auto.	com.	auto.	com.	auto.
Descriptive statistics [$corr_{sem}$]	M=.866 SD=.089	M=.983 SD=.023	M=.849 SD=.078	M=.964 SD=.016	M=.937 SD=.062	M=.981 SD=.034	M=.755 SD=.104	M=.945 SD=.038
Res. of Wilc. rank-sum tests	W=325.5 p=0.000 ****		W=237.0 p=0.000 ****		W=201.0 p=0.034 **		W=237.5 p=0.000 ****	
$\boldsymbol{H}_{A3}$: *The use of Automated Planning increases the semantic completeness of constructed process models.*								
	com.	auto.	com.	auto.	com.	auto.	com.	auto.
Descriptive statistics [$comp_{sem}$]	M=.895 SD=.067	M=.992 SD=.018	M=.899 SD=.045	M=.984 SD=.016	M=.957 SD=.065	M=.989 SD=.020	M=.947 SD=.043	M=.968 SD=.027
Res. of Wilc. rank-sum tests	W=325.0 p=0.000 ****		W=227.0 p=0.000 ****		W=178.0 p=0.169		W=163.0 p=0.177	
$\boldsymbol{H}_{B}$: *The use of Automated Planning decreases the required time for the modeling task.*								
	com.	auto.	com.	auto.	com.	auto.	com.	auto.
Descriptive statistics [minutes]	M=39.0 SD=5.75	M=23.7 SD=8.68	M=67.5 SD=8.21	M=46.6 SD=8.13	M=19.6 SD=6.57	M=14.1 SD=4.2	M=61.8 SD=11.24	M=51.1 SD=8.4
Results of the t-tests	t=-6.387; df=31.3 p=0.000 ****		t=-7.119; df=28.8 p=0.000 ****		t=-2.894; df=27.2 p=0.004 ***		t=-3.013; df=25.7 p=0.003 ***	

Table 4.3: The Influence of Automated Planning on the Task Performance of Process Modelers

*Significance codes: 0 '****' 0.001 '***' 0.01 '**' 0.05 '*' 0.1*

to higher initial setup cost. However, with our research, we have shown that not only the quality of the resulting, rather small process models (❶-❸) could be increased but also the required time for constructing the according modeling tasks could be decreased.

The participants are more experienced in using a common process modeling tool (i.e., ARIS Express) than in using Automated Planning as they used common process modeling tools in their prior studies. Thus, the results indicate that Automated Planning is advantageous compared to common process modeling tools.

4.1.5.3 Implications for Research and Practice

Our research illustrates the importance of research on the topic of how to support process modelers by means of algorithms. The approach at hand is mathematically proven to construct complete, correct, and minimal process models. However, the algorithms rely on a set of annotated actions. These annotations are not explicitly required when using a common process modeling approach. We have shown that using Automated Planning to construct process models from scratch or to adapt already annotated process models is beneficial. However, future work is required that enables annotating common process models as a basis for future adaptations by means of an Automated Planning approach. Some process mining works, for instance, already rely on rather similar formal foundations and are promising for further research. Verbeek et al. (2007) create variables from state spaces obtained through process mining by means of a Petri net synthesis technique based on so-called regions. Kindler et al. (2006) denote actions in terms of their input and output documents and describe states through the actions that were applied.

In practice, the efficiency of process modeling projects is a major issue. Our research shows that Automated Planning of process models has a positive influence on the task performance of process modelers. Hence, for transferring these findings into practical use, tool vendors can make use of our findings and the developed approaches. Further work has to be done to integrate the appropriate functionality in well-known process modeling tools. These features will enable modelers to avoid syntactic errors in process models. They further enable process modelers to construct semantically better process models in equal or less time.

Additionally, our findings indicate that novices are able to construct less erroneous process models in equal or less time. This opens plenty of chances from a managerial perspective. Most obviously, the number of required professional process modelers (and thus personnel costs) could potentially be reduced for business process modeling projects. Enabling novices to construct process models on their own further may lead to decentralized process modeling initiatives that could be conducted by one department

or even employee solely, which further leads to an improved end to end performance for such initiatives.

Further, if less dedicated staff is required, the demand for coordination and communication within a process modeling project could potentially be reduced. With less communication between participants in a process modeling project, the chances for misunderstandings that lead to erroneous process models could be reduced as well. Such implicit consequences have, so far, not been part of our evaluation.

4.1.5.4 Threats to Validity

In this Section, we want to discuss how far the results of our experiment support our claim about the influence of Automated Planning on the task performance of process modelers. Shadish et al. (2002)present different types of validity and common threats that we want to discuss regarding our research:

The **statistical conclusion validity** describes whether adequate statistical methods are used. We used Wilcoxon rank-sum tests which do not rely on normally distributed data. For the normally distributed classes, we relied on t-tests. Referring to literature (cf. Bandara et al., 2005; Mans et al., 2013; Overhage et al., 2012; Rosemann et al., 2001; Sedera et al., 2002), we expect having considered the relevant variables. However, as we described the Automated Planning approach in detail before the experiment, the way of the description might influence the results of the experiment. Further, the fact that all participants were aware of the other experimental group might be a confounding factor.

Internal validity refers to the question whether the difference in the task performance of process modelers is a consequence of using an Automated Planning approach or not. Here, the experiment was conducted once in one moment and place. Thus, threats caused by variance in implementation or selection of participants can be excluded. However, participants had to model two process models consecutively, which could cause learning or exhaustion effects. Thus, we randomized the order of the modeling tasks across the participants. Consequently, we could mitigate learning, maturation, exhaustion, and instrumentation effects. The modeling expertise of the participants was checked in the questionnaire.

Construct validity refers to the question, in how far our instruments reflect the constructs of the task performance of process modelers, personal factors, and process model factors correctly. We followed existing works and their proposals for using the quality of process models as "obtained outcomes" and required time for the modeling task as "invested resources" as factors for the task performance of process modelers (Bandara et al., 2006; Mans et al., 2013). However, due to the vague definition of "task performance of process modelers", the construct validity can hardly be assessed objectively.

Nonetheless, to mitigate the most obvious pitfalls, our instrumentation referred to multiple methods of assessing constructs that were pretested and used by other authors before (cf., e.g., Mendling et al., 2012b).

External validity describes how far the causal relations discussed in this experiment can be generalized over different settings. As discussed before, a wide range of factors (e.g., modeling purpose, modeling language) can influence the task performance of process modelers. However, they are not part of our experimental setup. We intentionally chose the form of a laboratory experiment to analyze the effect of using Automated Planning in isolation. The isolated analysis allows an interpretation whether our results are caused by the treatment or result from an incidental combination of other factors (see internal validity). However, this isolated analysis limits the generalizability of our results. Obviously, it takes further research in the field to analyze how strong the influence of Automated Planning is in more general settings.

4.1.6 Summary, Limitations, and Outlook

To the best of our knowledge, our study is the first, evaluating the influence of Automated Planning on the task performance of process modelers by empirical analysis. After introducing the theoretical foundation and our experimental design, we evaluate the influence of Automated Planning on the task performance of process modelers in terms of the required modeling time as well as the semantic and syntactic correctness and the semantic completeness of the resulting process models. The results show that Automated Planning significantly influences the task performance of modelers. However, our work has several limitations, which we discuss in the following and address in future work.

First to mention, the use of students in a class setting. Students can be considered as novices in process modeling (cf. Recker et al., 2012), a target group where especially the correctness and completeness of process models is a well-known issue, which makes them eligible candidates for our experiment. The students in our experiment represent a rather homogenous group, whereas, among practitioners, there might be more variability, especially concerning "modeling expertise". Hence, the results might be different when considering experienced practitioners. In order to evaluate this, additional research is required. Further, we did not verify the self-assessed personal factor "modeling expertise" during the experiment. This could potentially be achieved by taking the quality of the preliminary homework into consideration as an indicator for their modeling expertise in a next experiment. However, our aim was to provide a first indication in how far Automated Planning influences process modelers and hence, this possibly confounding aspect should be evaluated in more detail in further research as well.

Second, comparing two specific approaches, namely one particular Automated Planning approach and one particular common process modeling tool threatens the generalizability of our findings. As the participants have been using ARIS Express in prior classes, we rely on it as a common process modeling tool. We expect a lower task performance when using a so far unknown tool. Hence, the results of our experiment serve as a lower bound, and we expect the differences between two unknown modeling tools to be even greater. In this context, the experiment should be repeated with different common process modeling tools like, for instance, Signavio or Camunda Modeler[1] as well as different Automated Planning tools. Additionally, common process modeling tools have been improved over time, whereas the Automated Planning tool is a scientific prototype. Thus, revising and improving its user interface may improve the results. This should be addressed by further research on the usability of the Automated Planning tool at hand.

Third, we relied on four particular processes, partly of moderate size and complexity, though well known for the participants. However, according to Mendling et al. (2008), especially larger (they refer to 40 actions and more) and more complex process models likely contain more errors. Thus, our findings should also be verified for larger processes, which is part of future research. Additionally, we rely on two processes for each of the two cases of constructing process models from scratch and adapting them. In both cases, the processes differ considerably regarding their size and complexity. Thus, it is likely that the participants of the experiment assess the complexity of the processes as considerably different as well. Hence, in a second experiment, additional processes of medium size and complexity should be taken into account as well.

Additionally, with the research of Krause et al. (2013) in mind, assessing the economic value of Automated Planning is still an open topic. With our research, we provided a first basis for further research, as we addressed some well-known factors that affect the economic value of business process management projects (i.e., the required time for modeling tasks, influencing the costs, and the syntactic and semantic quality of resulting process models, influencing the value contribution of process modeling). However, based on the research of Krause et al. (2013) and our research, further work is necessary in order to gain detailed insights into the business value of Automated Planning. For instance, one could possibly find lower thresholds for the estimated size and complexity of process models to assess the economic reasonability of Automated Planning for a modeling task at hand. Lastly, other levels of quality should be evaluated as well. In our research, we focused on the semantic and the syntactic quality of process models.

[1] https://www.signavio.com/ resp. https://camunda.com/de/products/modeler/

However, it is still to evaluate, whether, for instance, the pragmatic quality (Krogstie et al., 2006) could be increased as well.

4.1.7 References

Aguilar-Savén, R. S. (2004). "Business process modelling: Review and framework." In: *International Journal of Production Economics* 90.2, pp. 129–149. ISSN: 09255273. DOI: 10.1016/S0925-5273(03)00102-6.

Bandara, W., Gable, G. G., and Rosemann, M. (2005). "Factors and measures of business process modelling: Model building through a multiple case study." In: *European Journal of Information Systems* 14.4, pp. 347–360. ISSN: 0960-085X. DOI: 10.1057/palgrave.ejis.3000546.

Bandara, W., Gable, G. G., and Rosemann, M. (2006). "Business process modeling success: An empirically tested measurement model." In: *Proceedings of the 27th International Conference on Information Systems*. Ed. by D. Straub and S. Klein. Milwaukee, Wisconsin: University of Wisconsin, pp. 1–20. URL: http://eprints.qut.edu.au/11131/.

Becker, J., Algermissen, L., Falk, T., Pfeiffer, D., and Fuchs, P. (July 2006). "Model Based Identification and Measurement of Reorganization Potential in Public Administrations – the PICTURE-Approach." In: *Proceedings of the 10th Pacific Asia Conference on Information Systems*. Kuala Lumpur, Malaysia, pp. 860–875. URL: http://www.pacis-net.org/file/2006/1148.pdf.

Becker, J., Pfeiffer, D., Räckers, M., Falk, T., and Czerwonka, M. (2015). "Semantic Business Process Modelling and Analysis." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 187–217. ISBN: 978-3-642-45099-0. DOI: 10.1007/978-3-642-45100-3_9.

Becker, J., Rosemann, M., and von Uthmann, C. (2000). "Guidelines of Business Process Modeling." In: *Business Process Management*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, W. M. P. van der Aalst, J. Desel, and A. Oberweis. Vol. 1806. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 30–49. ISBN: 978-3-540-67454-2. DOI: 10.1007/3-540-45594-9_3.

Becker, J., Thome, I., Weiß, B., and Winkelmann, A. (July 2010a). "Constructing a Semantic Business Process Modelling Language for the Banking Sector: An Evolutionary Dyadic Design Science Approach." In: *Enterprise Modelling and Information Systems Architectures* 5.1, pp. 4–25. DOI: 10.18417/emisa.5.1.1. URL: https://www.emisa-journal.org/emisa/article/view/64.

Becker, J., Weiß, B., and Winkelmann, A. (2010b). "Utility vs. Efforts of Business Process Modeling — An Exploratory Survey in the Financial Sector." In: *Proceedings of the Multikonferenz Wirtschaftsinformatik 2010*. Ed. by M. Schumann, L. M. Kolbe, M. H. Bre-

itner, and A. Frerichs. Göttingen: Universitätsverlag Göttingen, pp. 41–54. ISBN: 978-3-941875-31-9. URL: http://udoo.uni-muenster.de/downloads/publications/2361.pdf.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Betz, S., Klink, S., Koschmider, A., and Oberweis, A. (2006). "Automatic user support for business process modeling." In: *Proceedings of the 3rd European Semantic Web Conference (ESWC 2006)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.83.4899.

Branco, M. C., Xiong, Y., Czarnecki, K., Küster, J., and Völzer, H. (2014). "A case study on consistency management of business and IT process models in banking." In: *Software & Systems Modeling* 13.3, pp. 913–940. ISSN: 1619-1366. DOI: 10.1007/s10270-013-0318-8.

Brockmans, S., Ehrig, M., Koschmider, A., Oberweis, A., and Studer, R. (2006). "Semantic Alignment Of Business Processes." In: *Proceedings of the Eighth International Conference on Enterprise Information Systems (ICEIS 2006)*, pp. 191–196.

Ding, Z., Sun, Y., Liu, J., Pan, M., and Liu, J. (2015). "A genetic algorithm based approach to transactional and QoS-aware service selection." In: *Enterprise Information Systems*, pp. 1–20. ISSN: 1751-7575. DOI: 10.1080/17517575.2015.1048832.

Fahland, D., Favre, C., Koehler, J., Lohmann, N., Völzer, H., and Wolf, K. (2011). "Analysis on demand: Instantaneous soundness checking of industrial business process models." In: *Data & Knowledge Engineering* 70.5, pp. 448–466. DOI: 10.1016/j.datak.2011.01.004.

Fellmann, M., Delfmann, P., Koschmider, A., Laue, R., Leopold, H., and Schoknecht, A. (2015a). "Semantic Technology in Business Process Modeling and Analysis: Part 1: Matching, Modeling Support, Correctness and Compliance." In: *Emisa Forum* 35.1.

Fleischmann, A., Kannengiesser, U., Schmidt, W., and Stary, C. (2013). "Subject-Oriented Modeling and Execution of Multi-agent Business Processes." In: *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*. IEEE, pp. 138–145. ISBN: 978-0-7695-5145-6. DOI: 10.1109/WI-IAT.2013.102.

Frederiks, P. and van der Weide, T. (2006). "Information modeling: The process and the required competencies of its participants." In: *Data & Knowledge Engineering* 58.1, pp. 4–20. DOI: 10.1016/j.datak.2005.05.007.

Gordijn, J., Akkermans, H., and van Vliet, H. (2000). "Business Modelling Is Not Process Modelling." In: *Conceptual Modeling for E-Business and the Web*. Ed. by G. Goos,

J. Hartmanis, J. van Leeuwen, S. W. Liddle, H. C. Mayr, and B. Thalheim. Vol. 1921. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 40–51. ISBN: 978-3-540-41073-7. DOI: 10.1007/3-540-45394-6_5.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems* (*ICIS*). Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., Lewerenz, L., and Zimmermann, S. (Mar. 2015a). "Quality of Service-aware Service Selection: A Novel Approach Considering Potential Service Failures and Non-Deterministic Service Values." In: *INFORMS Service Science, A Journal of the Institute for Operations Research and the Management Sciences* 7.1, pp. 48–69. DOI: 10.1287/serv.2015.0093.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik* (*MKWI*). Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hepp, M. and Dumitri, R. (2007). "An Ontology Framework for Semantic Business Process Management." In: *Proceedings of the 8th International Conference on Business Informatics* (*WI 2007*): *eOrganisation: Service-*, pp. 423–440.

Hepp, M., Leymann, F., Bussler, C., Domingue, J., Wahler, A., and Fensel, D. (2005). "Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management." In: *IEEE International Conference on E-Business Engineering* (*ICEBE 2005*) 0, pp. 535–540. DOI: 10.1109/ICEBE.2005.110.

Hoffmann, J. and Brafman, R. I. (2005). "Contingent Planning via Heuristic Forward Search with Implicit Belief States." In: *Proceedings of the 15th International Conference on Automated Planning and Scheduling*. ICAPS'05. AAAI Press, pp. 71–88. ISBN: 1-57735-220-3.

Hoffmann, J., Weber, I., and Kraft, F. M. (July 2012). "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management." In: *Journal of Artificial Intelligence Research* 44.1, pp. 587–632. DOI: 10.1613/jair.3636.

Hoppenbrouwers, S. J. B. A., Proper, H. A., and van der Weide, T. P. (2005). "A Fundamental View on the Process of Conceptual Modeling." In: *Conceptual Modeling – ER 2005*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, L. Delcambre, C. Kop, H. C. Mayr, J. Mylopoulos, and O. Pastor. Vol. 3716. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 128–143. ISBN: 978-3-540-29389-7. DOI: 10.1007/11568322_9.

Hornung, T., Koschmider, A., and Lausen, G. (2008). "Recommendation Based Process Modeling Support: Method and User Experience." In: *Conceptual Modeling - ER 2008*. Ed. by Q. Li, S. Spaccapietra, E. Yu, and A. Olivé. Vol. 5231. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 265–278. ISBN: 978-3-540-87876-6. DOI: 10.1007/978-3-540-87877-3_20.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference on Advanced Information Systems Engineering (CAiSE)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

Indulska, M., Recker, J., Rosemann, M., and Green, P. (2009). "Business Process Modeling: Current Issues and Future Challenges." In: *Advanced Information Systems Engineering*. Ed. by P. van Eck, J. Gordijn, and R. Wieringa. Vol. 5565. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 501–514. ISBN: 978-3-642-02143-5. DOI: 10.1007/978-3-642-02144-2_39.

Jedlitschka, A., Ciolkowski, M., and Pfahl, D. (2008). "Reporting Experiments in Software Engineering." In: *Guide to advanced empirical software engineering*. Ed. by F. Shull, J. Singer, and D. I. K. Sjøberg. London: Springer, pp. 201–228. ISBN: 978-1-84800-044-5.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security (IJCSIS)* 7.2, pp. 147–152.

Kindler, E., Rubin, V., and Schäfer, W. (2006). "Process Mining and Petri Net Synthesis." In: *Business Process Management Workshops*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Eder, and S. Dustdar. Vol. 4103. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 105–116. ISBN: 978-3-540-38444-1. DOI: 10.1007/11837862_12.

Kitchenham, B. A., Pfleeger, S. L., Pickard, L. M., Jones, P. W., Hoaglin, D. C., El Emam, K., and Rosenberg, J. (2002). "Preliminary guidelines for empirical research in software engineering." In: *Software Engineering, IEEE Transactions on* 28.8, pp. 721–734. ISSN: 0098-5589. DOI: 10.1109/TSE.2002.1027796.

Krause, F., Bewernik, M.-A., and Fridgen, G. (2013). "Valuation of Manual and Automated Process Redesign from a Business Perspective." In: *Business Process Management Journal* 19.1.

Krogstie, J., Sindre, G., and Jørgensen, H. (2006). "Process models representing knowledge for action: A revised quality framework." In: *European Journal of Information Systems* 15.1, pp. 91–102. ISSN: 0960-085X. DOI: 10.1057/palgrave.ejis.3000598.

Kusiak, A., Nick Larson, T., and Wang, J. (1994). "Reengineering of design and manufacturing processes." In: *Computers & Industrial Engineering* 26.3, pp. 521–536. ISSN: 03608352. DOI: 10.1016/0360-8352(94)90048-5.

La Rosa, M., ter Hofstede, A. H. M., Wohed, P., Reijers, H. A., Mendling, J., and van der Aalst, W. M. P. (2011c). "Managing Process Model Complexity via Concrete Syntax Modifications." In: *IEEE Transactions on Industrial Informatics* 7.2, pp. 255–265. DOI: 10.1109/TII.2011.2124467.

Lautenbacher, F., Eisenbarth, T., and Bauer, B. (2009). "Process model adaptation using semantic technologies." In: *2009 13th Enterprise Distributed Object Computing Conference Workshops, EDOCW*, pp. 301–309. DOI: 10.1109/EDOCW.2009.5331985.

Leopold, H., Mendling, J., and Gunther, O. (2016). "Learning from Quality Issues of BPMN Models from Industry." In: *IEEE Software* 33.4, pp. 26–33. ISSN: 0740-7459. DOI: 10.1109/MS.2015.81.

Lindland, O. I., Sindre, G., and Solvberg, A. (1994). "Understanding quality in conceptual modeling." In: *IEEE Software* 11.2, pp. 42–49. ISSN: 0740-7459. DOI: 10.1109/52.268955.

Mans, R., Reijers, H. A., Berends, H., Bandara, W., and Prince, R. (2013). "Business Process Mining Success." In: *Proceedings of the 21st European Conference on Information Systems*.

Mendling, J., Neumann, G., and van der Aalst, W. M. P. (2007a). "Understanding the Occurrence of Errors in Process Models Based on Metrics." In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, and Z. Tari. Vol. 4803. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 113–130. ISBN: 978-3-540-76846-3. DOI: 10.1007/978-3-540-76848-7_9.

Mendling, J., Reijers, H. A., and van der Aalst, W. M. P. (2010). "Seven process modeling guidelines (7PMG)." In: *Information and Software Technology* 52.2, pp. 127–136. ISSN: 0950-5849.

Mendling, J., Sánchez-González, L., García, F., and La Rosa, M. (2012a). "Thresholds for error probability measures of business process models." In: *Journal of Systems and Software* 85.5, pp. 1188–1197. DOI: 10.1016/j.jss.2012.01.017.

Mendling, J., Strembeck, M., and Recker, J. (2012b). "Factors of process model comprehension—Findings from a series of experiments." In: *Decision Support Systems* 53.1, pp. 195–206. DOI: 10.1016/j.dss.2011.12.013.

Mendling, J., Verbeek, H. M. W., van Dongen, B. F., van der Aalst, W. M. P., and Neumann, G. (2008). "Detection and prediction of errors in EPCs of the SAP reference model." In: *Data & Knowledge Engineering* 64.1, pp. 312–329. DOI: 10.1016/j.datak.2007.06.019.

Moreno-Montes de Oca, I., Snoeck, M., Reijers, H. A., and Rodríguez-Morffi, A. (2015). "A systematic literature review of studies on business process modeling quality." In: *Information and Software Technology* 58, pp. 187–205. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2014.07.011.

OMG Unified Modeling Language TM (OMG UML): Version 2.5 (2015). URL: http://www.omg.org/spec/UML/2.5 (visited on 05/04/2016).

Overhage, S., Birkmeier, D. Q., and Schlauderer, S. (2012). "Quality Marks, Metrics, and Measurement Procedures for Business Process Models." In: *Business & Information Systems Engineering* 4.5, pp. 229–246. ISSN: 1867-0202. DOI: 10.1007/s12599-012-0230-8.

Paik, I., Chen, W., and Huhns, M. N. (2014). "A scalable architecture for automatic service composition." In: *Services Computing, IEEE Transactions on* 7.1, pp. 82–95.

Prat, N., Comyn-Wattiau, I., and Akoka, J. (2015). "A Taxonomy of Evaluation Methods for Information Systems Artifacts." In: *Journal of Management Information Systems* 32.3, pp. 229–267. DOI: 10.1080/07421222.2015.1099390.

Recker, J., Safrudin, N., and Rosemann, M. (2012). "How novices design business processes." In: *Information Systems* 37.6, pp. 557–573. ISSN: 0306-4379. DOI: 10.1016/j.is.2011.07.001.

Rosemann, M. (2006). "Potential pitfalls of process modeling: part A." In: *Business Process Management Journal* 12.2, pp. 249–254. DOI: 10.1108/14637150610657567. URL: http://www.ingentaconnect.com/content/mcb/157/2006/00000012/00000002/art00009.

Rosemann, M., Sedera, W., and Gable, G. G. (2001). "Critical success factors of process modeling for enterprise systems." In: *7th Americas Conference on Information Sys-*

tems. Boston, MA: Association for Information Systems, pp. 1128–1130. URL: http://eprints.qut.edu.au/25119/.

Rosemann, M. and vom Brocke, J. (2015). "The Six Core Elements of Business Process Management." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Heidelberg Dordrecht London New York: Springer, pp. 107–122.

Roy, S., Sajeev, A. S. M., Bihary, S., and Ranjan, A. (2014). "An Empirical Study of Error Patterns in Industrial Business Process Models." In: *IEEE Transactions on Services Computing* 7.2, pp. 140–153. ISSN: 1939-1374. DOI: 10.1109/TSC.2013.10.

Russell, N., ter Hofstede, A. H. M., and Mulyar, N. (2006a). "Workflow ControlFlow Patterns: A Revised View." In: *BPM Center Report BPM-06-22*. URL: http://bpmcenter.org/reports.

Russell, N., van der Aalst, W. M. P., and Ter Hofstede, A. (2016). *Workflow patterns: The definitive guide*. Information systems. Cambridge, Massachusetts: The MIT Press. ISBN: 978-0-262-02982-7.

Scheer, A.-W. and Brabänder, E. (2010). "The Process of Business Process Management." In: *Handbook on Business Process Management 2*. Ed. by J. vom Brocke and M. Rosemann. Springer-Verlag Berlin Heidelberg, pp. 239–266.

Scholtz, B., Calitz, A., and Snyman, I. (2013). "The usability of collaborative tools." In: *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference on - SAICSIT '13*. Ed. by J. McNeill, K. Bradshaw, P. Machanick, and M. Tsietsi. New York, New York, USA: ACM Press, p. 347. ISBN: 9781450321129. DOI: 10.1145/2513456.2513503.

Sedera, W., Rosemann, M., and Gable, G. G. (2002). "Measuring Process Modelling Success." In: *Proceedings of the 10th European Conference on Information Systems*, pp. 331–341.

Seethamraju, R. and Marjanovic, O. (2009). "Role of process knowledge in business process improvement methodology: A case study." In: *Business Process Mgmt Journal* 15.6, pp. 920–936. ISSN: 1463-7154. DOI: 10.1108/14637150911003784.

Shadish, W. R., Cook, T. D., and Campbell, D. T. (2002). *Experimental and quasi-experimental designs for generalized causal inference*. Belmont, CA: Wadsworth Cengage Learning. ISBN: 978-0395615560.

Soffer, P., Kaner, M., and Wand, Y. (2012). "Towards Understanding the Process of Process Modeling: Theoretical and Empirical Considerations." In: *Business Process Management Workshops*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, F. Daniel, K. Barkaoui, and S. Dustdar. Vol. 99. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 357–369. ISBN: 978-3-642-28107-5. DOI: 10.1007/978-3-642-28108-2_35.

Thomas, O. and Fellmann, M. (2009). "Semantic Process Modeling – Design and Implementation of an Ontology-based Representation of Business Processes." In: *Busi-*

ness & Information Systems Engineering 1.6, pp. 438–451. ISSN: 1867-0202. DOI: 10.1007/s12599-009-0078-8.

Van der Aalst, W. M. P. (2013). "Business process management: A comprehensive survey." In: *ISRN Software Engineering* 2013.

Van der Aalst, W. M. P. and Jablonski, S. (2000b). "Dealing with workflow change: identification of issues and solutions." In: *International Journal of Computer Systems Science & Engineering* 15.5, pp. 267–276.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Verbeek, H. M. W., Pretorius, A. J., van der Aalst, W. M. P., and van Wijk, J. J. (2007). "Visualizing state spaces with Petri nets." In: *Computer Science Report* 7.01.

Vom Brocke, J., Schmiedel, T., Recker, J., Trkman, P., Mertens, W., and Viaene, S. (2014). "Ten principles of good business process management." In: *Business Process Management Journal* 20.4, pp. 530–548. ISSN: 1463-7154. DOI: 10.1108/BPMJ-06-2013-0074.

Wang, P., Ding, Z., Jiang, C., and Zhou, M. (2014). "Automated web service composition supporting conditional branch structures." In: *Enterprise Information Systems* 8.1, pp. 121–146. ISSN: 1751-7575. DOI: 10.1080/17517575.2011.584132.

Wang, W. and Brooks, R. J. (2007). "Empirical investigations of conceptual modeling and the modeling process." In: *Proceedings of the 2007 Winter Simulation Conference*. Ed. by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, pp. 762–770. DOI: 10.1109/WSC.2007.4419671.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Weber, I., Hoffmann, J., and Mendling, J. (2008b). "Semantic business process validation." In: *Proceedings of the 3rd International Workshop on Semantic Business Process Management*.

Wetzstein, B., Ma, Z., Filipowska, A., Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., and Cicurel, L. (2007). "Semantic Business Process Management: A Lifecycle Based Requirements Analysis." In: *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007) in conjunction with the 3rd European Semantic Web Conference (ESWC 2007)*.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Berlin and New York: Springer. ISBN: 978-3-642-29043-5.

Zamperoni, A. and Löhr-Richter, P. (1993). "Enhancing the Quality of Conceptual Database Specifications through Validation." In: *Proceedings of the 12th International Conference on Entity-Relationship Approach (ER'93)*, pp. 87–99.

Zur Muehlen, M. and Rosemann, M. (2004). "Multi-Paradigm Process Management." In: *Proceedings of CAiSE'04 Workshops - 5th Workshop on Business Process Modeling, Development and Support (BPMDS 2004)*. Ed. by J. Grundspenkis and M. Kirikova, pp. 169–175.

5
Conclusion

In Section 5.1, the major findings of this dissertation are summarized. Thereafter, in Section 5.2, possible aspects for future work, based on the limitations of this dissertation, are discussed. The points that are identified within this Chapter do not address single papers presented in this dissertation but address the dissertation as a whole, since respective summaries and outlooks are contained directly within the papers.

5.1 Major Findings

Today's business world that "changes faster all the time" (Harmon, 2019, p. 441) is expecting companies to be more and more flexible and agile. BPM, in this context, not only serves as enabler for flexibility and agility but may, to some extent, be an impediment as well, as there is a huge amount of manual efforts associated with BPM initiatives (Bowers et al., 1995; Škrinjar et al., 2010). Hence, "BPM should make opportune use of technology" (vom Brocke et al., 2014, p. 533) to reduce those manual efforts. In particular, process modeling is a time consuming and therefore costly task in practice (cf., e.g., Becker et al., 2010b; Hornung et al., 2007). On the one hand, the research field Automated Planning of Process Models strives this issue by decreasing the manual efforts while on the other hand, the flexibility by definition (cf. van der Aalst, 2013) of processes could be increased. This dissertation contributes to this research area and extends it by particular quantitative approaches to further decrease the manual efforts and to increase the flexibility by definition of processes. The main contributions from both of these focal topics are as follows.

Concerning the first focal topic, FT1 decreasing the manual efforts for constructing to-be process models, firstly an approach to construct the control flow pattern simple merge is presented in Paper 1 striving RQ1.1. The approach to automatically construct the control flow pattern simple merge as presented in Paper 1 constructs minimal and complete process models and is able to merge multiple, nested exclusive choices as well as exclusive choices within parallelization compounds. In addition to that, the approach constructs so-called pattern compounds (cf., e.g., La Rosa et al., 2011c) that increase the readability and understandability of process models. Secondly, an experimental study that answers the question whether and in which particular situations Automated Planning of Process Models is beneficial in contrast to process modeling using regular, well-known tools in Paper 5 answering RQ3 is presented. Paper 5 particularly presents a laboratory experiment that shows that Automated Planning of Process Models is beneficial with respect to the task performance of process modelers. The laboratory experiment shows that the invested resources in terms of the required modeling time for modeling tasks could be decreased, especially when constructing larger process models. Further,

it was shown that the outcome of process modeling tasks in terms of the quality of the resulting process models could be increased by relying on Automated Planning.

Regarding the second focal topic, **FT2** increasing the flexibility of to-be processes, three selected factors IF1 to IF3 that require processes to be flexible are considered in the dissertation at hand. Paper 2 strives RQ1.2 and addresses IF2 exogenous non-static context influencing processes by means of Automated Planning of Process Models. In particular, the presented approach extends an existing set-theoretic planning domain by so-called context variables and including context-related information in the preconditions of actions. Further, to consider context as non-static, so-called context-signals and receive context actions are introduced to handle changing context. Finally, an existing algorithm from the field of Automated Planning of Process Models is extended to enable the planning of context-aware process models. Paper 3 aims at at answering RQ1.3 coordinating IF3 multiple contributing actors by means of an Automated Planning approach. To do so, an existing set-theoretic planning domain is extended to consider actor-specific information. Further, the concept of partnerships, meaning the fact that actions may potentially need to be conducted by a certain number of actors, is included in the planning domain. To conclude, an existing planning algorithm is extended to construct so-called join actions resp. split actions to build (resp. disband) partnerships of actors throughout a process. Paper 4 addresses RQ2 and deals with the issue of adapting existing process models to IF1 upcoming needs for change by means of Automated Planning of Process Models. Initially, a structured analysis of potentially occurring changes is conducted, based on the planning domain. Thereafter, approaches for the adaptation of process graphs resulting from altering one particular element in the planning domain are presented. The combined approach to adapt process graphs to upcoming needs for change is thereafter extensively evaluated by means of mathematical proofs, a case study as well as a simulation experiment.

All approaches share a common formal foundation, namely a common planning domain to guarantee a maximum of compatibility with existing approaches (cf., e.g., Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2009; Heinrich et al., 2019; Meyer et al., 2006; Sycara et al., 2003). Moreover, each of the papers comprises novel concepts as well as a particular algorithm. The key properties of each approach are formally verified. Further, the approaches are prototypically implemented and evaluated in different real-world scenarios. In summary, all presented approaches proposed in the dissertation expand the research area of Automated Planning of Process Models. Hence, the dissertation provides particular novel approaches in two focal topics that support organizations, business analysts and especially process modelers in increasing their agility for facing the challenges of todays fast-paced business world. However, there remains a vast amount of interesting directions for further research.

5.2 Outlook & Future Work

This dissertation has focused on five particular research questions to address relevant issues in the field of Automated Planning of Process Models or BPM in general. However, today's fast-changing business world and the resulting pressure on companies requires processes to be more flexible, and a reduction of efforts in the field of BPM beyond the extent of what this work was able to achieve. Some possible directions for future work shall be outlined in the remainder.

To begin with, the presented approaches in this dissertation share a common formal foundation in form of a nondeterministic state-transition system. Relying on such a nondeterministic state-transition system guarantees compatibility with existing approaches (cf., e.g., Bertoli et al., 2001; Bertoli et al., 2006; Heinrich et al., 2009; Heinrich et al., 2019; Meyer et al., 2006; Sycara et al., 2003) at least to some extent. However, particular issues such as the construction of control flow patterns have so far been addressed in isolation from each other. Even though, for instance, the proposed approach to construct the control flow pattern simple merge (cf. Section 2.1) makes it possible to merge multiple nested exclusive choices as well as exclusive choices in parallelization compounds, a holistic, integrated approach that is capable of constructing process models comprising all basic control flow patterns (cf. Russell et al., 2006a; Russell et al., 2016; van der Aalst et al., 2003) is still an unsolved issue. This issue is still non-trivial (i.e., not only correctly ordering the execution of the individual algorithms) as existing methods partly rely on a planning domain without control flow patterns as input for the according algorithms (cf., e.g., Heinrich et al., 2009; Heinrich et al., 2015b; Heinrich et al., 2019). However, such a holistic, integrated approach is of particular interest as it lays the foundation for a fruitful use in practice and hence to fully exploit the potential of Automated Planning of Process Models in practical use. Integrating such a realized, integrated approach into market-proven PAIS is a logical next step. For instance, the tremendous success of Celonis as a commercial provider of Process Mining software allows surmising the potential for approaches that incorporate automation in BPM. Hence, integrating Automated Planning of Process Models into tools such as Signavio, Camunda, or the ARIS suite seems promising to enable exploiting its potential in a business context.

Lastly, when talking about this range of topics, a field evaluation should be carried out in future work, either by relying on an integrated but scientific prototype or by means of a PAIS that integrates Automated Planning of Process Models. So far, no explicit field evaluation has been conducted. The economic evaluation, as presented in Section 4.1, relies on students and employs a laboratory experiment. However, to assess the economic impact of Automated Planning in business scenarios, field studies probably

would give exciting insights into how professionals interact with Automated Planing and whether it is economically beneficial in real-world projects as well.

Besides that, not only integrating existing approaches to construct basic control flow patterns seems beneficial but also the automated construction of additional, basic (cf. van der Aalst et al., 2003) or advanced control flow patterns (cf. van der Aalst et al., 2000a). In particular, "structural" (van der Aalst et al., 2003) control flow patterns such as arbitrary cycles are still not explicitly covered in the field of Automated Planning of Process Models. Even though some approaches (cf., e.g., Heinrich et al., 2008; Heinrich et al., 2012) are implicitly capable of coping with cycles - the presented approach to construct the control flow pattern simple merge is an example for that as well - other works do explicitly note that they are not able to cope with cycles (cf., e.g., Heinrich et al., 2019). Hence, further research on additional control flow patterns and especially on arbitrary cycles is promising as real-world processes oftentimes comprise cycles.

Moreover, approaches for the Automated Planning of context-aware processes (cf. Section 2.2) as well as for the coordination of multiple actors (cf. Section 2.3) have been presented in this dissertation. Other research areas such as (web) service selection are already aiming at comparable issues but employ integrated solutions. Bortlik et al. (2018), for instance, strive to resolve the issue of determining feasible service compositions for multiple actors, each of whom has own preferences and constraints, considering context information such as daytime or location. Hence, it seems promising to integrate both approaches and thereby to enable the Automated Planning of context-aware multi-actor process models, accordingly.

Combining the presented approaches with other works from related topics might provide fruitful outcomes as well. There are plenty possibilities to combine the presented approaches with other approaches in the general area of Business Process Automation. For instance, a combination with the aforementioned approach, presented by Bortlik et al. (2018) or related ones from the field of (web) service selection (e.g., Falou et al., 2009; Fujii et al., 2009; Lewerenz, 2015; Mehandjiev et al., 2012; Zhou et al., 2011) and execution may possibly yield interesting possibilities to enable automated planning, implementation, and lastly execution of context-aware multi-actor processes. As mentioned in Section 1.3, there exist several types of process flexibility, and a combination of multiple approaches that increase complementary types of flexibility may be advantageous. Hence, instead of only increasing the flexibility of definition of processes as presented in the dissertation, increasing the flexibility by deviation simultaneously could provide an interesting foundation for future research. Especially regarding context-aware processes, there might be the fact that particular context events have not been considered by modelers in advance and hence the flexibility by definition of the according process is still limited. Leveraging this issue with approaches that enable deviation from the pro-

cess model during process execution may result in a more comprehensive support for context-aware processes. Furthermore, the development of approaches to derive new process paths from these deviations as input for an automated adaptation of the process model can be an interesting field for future research.

Another possible direction for future research is the integration of concepts of other planning approaches into the Automated Planning of Process Models. For instance, probabilistic planning as well as temporal planning might provide fruitful extensions to the planning domain on that the approaches presented in the dissertation rely. Probabilistic planning extends the given nondeterministic planning domain by means of probabilities with which effects of actions occur. Temporal planning extends the given nondeterministic planning domain by means of information about the duration of conducting an action. Both extensions could potentially provide interesting benefits with respect to adaptive processes. For instance, effects that result from the conduction of actions may potentially come into force directly at starting to conduct the according action, with finishing the conduction or at some time in between. This idea is not new and existing works relying on temporal planning already cope with such so-called "intermediate effects" by relying on so-called auxiliary actions that break up an action into a sequence of ordered sub-actions (cf., e.g., Cimatti et al., 2018; Smith et al., 2008; Valentini et al., 2019), for example. While the integration of such concepts in the Automated Planning of Process Models is promising, it is still an unsolved topic. With concepts from probabilistic planning (cf. Bonet et al., 2001; Ghallab et al., 2004; Ghallab et al., 2016), a two-staged planning process could potentially be created. In the first step, process models could be constructed by means of Automated Planning of Process Models as presented in the dissertation. In a second step, the nondeterministic process model could then be refined by integrating probabilistic information with respect to the effects of actions or probabilities of the occurrence of context-events. For instance, context-aware process models could first be planned without probabilistic information and, shortly before implementation and execution, information such as the probability for changing weather could be taken into account.

To conclude, this dissertation forms a solid base for various interesting future research as Automated Planning of Process Models does not only provide an economical advantage when constructing and adapting process models but also increases the flexibility of processes. The dissertation itself has extended this research field by means of specific approaches supporting process modelers and business analysts, respectively organizations in general. However, significant challenges remain to be addressed and the need for research in this field is ongoing.

5.3 References

Becker, J., Weiß, B., and Winkelmann, A. (2010b). "Utility vs. Efforts of Business Process Modeling — An Exploratory Survey in the Financial Sector." In: *Proceedings of the Multikonferenz Wirtschaftsinformatik 2010*. Ed. by M. Schumann, L. M. Kolbe, M. H. Breitner, and A. Frerichs. Göttingen: Universitätsverlag Göttingen, pp. 41–54. ISBN: 978-3-941875-31-9. URL: http://udoo.uni-muenster.de/downloads/publications/2361.pdf.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bonet, B. and Geffner, H. E. (2001). "GPT: A Tool for Planning with Uncertainty and Partial Information." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pp. 82–87.

Bortlik, M., Heinrich, B., and Mayer, M. (Oct. 2018). "Multi User Context-Aware Service Selection for Mobile Environments." In: *Business & Information Systems Engineering* 60.5, pp. 415–430. DOI: 10.1007/s12599-018-0552-2.

Bowers, J., Button, G., and Sharrock, W. (1995). "Workflow From Within and Without: Technology and Cooperative Work on the Print Industry Shopfloor." In: *Proceedings of the Fourth European Conference on Computer-Supported Cooperative Work (ECSCW '95)*. Ed. by H. Marmolin, Y. Sundblad, and K. Schmidt. Dordrecht: Springer Netherlands, pp. 51–66. ISBN: 978-94-011-0349-7. DOI: 10.1007/978-94-011-0349-7_4.

Cimatti, A., Do, M., Micheli, A., Roveri, M., and Smith, D. E. (2018). "Strong temporal planning with uncontrollable durations." In: *Artificial Intelligence* 256, pp. 1–34. ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2017.11.006. URL: http://www.sciencedirect.com/science/article/pii/S0004370217301431.

Falou, M. E., Bouzid, M., Mouaddib, A.-I., and Vidal, T. (2009). "Automated Web Service Composition: A Decentralised Multi-agent Approach." In: *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pp. 387–394. DOI: 10.1109/WI-IAT.2009.68.

Fujii, K. and Suda, T. (2009). "Semantics-based context-aware dynamic service composition." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 4.2, p. 12. ISSN: 1556-4665.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Ghallab, M., Nau, D. S., and Traverso, P. (2016). *Automated Planning and Acting*. New York, NY: Cambridge University Press. ISBN: 978-1107037274.

Harmon, P. (2019). *Business Process Change - A Business Process Management Guide for Managers and Process Professionals*. Fourth Edition. Morgan Kaufmann.

Heinrich, B., Bewernik, M.-A., Henneberger, M., Krammer, A., and Lautenbacher, F. (2008). "SEMPA - A Semantic Business Process Management Approach for the Planning of Process Models." In: *Business & Information Systems Engineering* (*formerly Wirtschaftsinformatik*) 50.6, 445–460 (in German). URL: http://epub.uni-regensburg.de/23173/.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems* (*ICIS*). Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Heinrich, B., Krause, F., and Schiller, A. (2019). "Automated planning of process models: The construction of parallel splits and synchronizations." In: *Decision Support Systems* 125, p. 113096. ISSN: 0167-9236. DOI: 10.1016/j.dss.2019.113096. URL: http://www.sciencedirect.com/science/article/pii/S0167923619301253.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference on Advanced Information Systems Engineering* (*CAiSE*). URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

La Rosa, M., ter Hofstede, A. H. M., Wohed, P., Reijers, H. A., Mendling, J., and van der Aalst, W. M. P. (2011c). "Managing Process Model Complexity via Concrete Syntax Modifications." In: *IEEE Transactions on Industrial Informatics* 7.2, pp. 255–265. DOI: 10.1109/TII.2011.2124467.

Lewerenz, L. (2015). "A Heuristic Technique for an Efficient Decision Support in Context-aware Service Selection." In: *Proceedings of the 36th International Conference on Information Systems (ICIS)*, pp. 1–20. URL: https://epub.uni-regensburg.de/32660/.

Mehandjiev, N., Lécué, F., Carpenter, M., and Rabhi, F. A. (2012). "Cooperative Service Composition." In: *Advanced Information Systems Engineering*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Ralyté, X. Franch, S. Brinkkemper, and S. Wrycza. Vol. 7328. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 111–126. ISBN: 978-3-642-31094-2. DOI: 10.1007/978-3-642-31095-9_8.

Meyer, H. and Weske, M. (2006). "Automated service composition using heuristic search." In: *Business Process Management*, pp. 81–96.

Russell, N., ter Hofstede, A. H. M., and Mulyar, N. (2006a). "Workflow ControlFlow Patterns: A Revised View." In: *BPM Center Report BPM-06-22*. URL: http://bpmcenter.org/reports.

Russell, N., van der Aalst, W. M. P., and Ter Hofstede, A. (2016). *Workflow patterns: The definitive guide*. Information systems. Cambridge, Massachusetts: The MIT Press. ISBN: 978-0-262-02982-7.

Škrinjar, R., Vukšić, V., and Štemberger, M. (2010). "Adoption of Business Process Orientation Practices: Slovenian and Croatian Survey." In: *Business Systems Research* 1.1-2, pp. 5–19. ISSN: 1847-9375. DOI: 10.2478/v10305-012-0022-0.

Smith, D. E., Frank, J., and Cushing, W. (2008). "The ANML language." In: *The ICAPS-08 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Sycara, K., Paolucci, M., Ankolekar, A., and Srinivasan, N. (2003). "Automated discovery, interaction and composition of semantic web services." In: *Web Semantics: Science, Services and Agents on the World Wide Web* 1.1, pp. 27–46. ISSN: 1570-8268.

Valentini, A., Micheli, A., and Cimatti, A. (2019). *Temporal Planning with Intermediate Conditions and Effects*. arXiv: 1909.11581 [cs.AI].

Van der Aalst, W. M. P. (2013). "Business process management: A comprehensive survey." In: *ISRN Software Engineering* 2013.

Van der Aalst, W. M. P., Barros, A. P., ter Hofstede, A. H. M., and Kiepuszewski, B. (2000a). "Advanced Workflow Patterns." In: *Cooperative Information Systems*. Ed. by P. Scheuermann and O. Etzion. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 18–29. ISBN: 978-3-540-45266-9.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Vom Brocke, J., Schmiedel, T., Recker, J., Trkman, P., Mertens, W., and Viaene, S. (2014). "Ten principles of good business process management." In: *Business Process Management Journal* 20.4, pp. 530–548. ISSN: 1463-7154. DOI: 10.1108/BPMJ-06-2013-0074.

Zhou, J., Gilman, E., Palola, J., Riekki, J., Ylianttila, M., and Sun, J. (2011). "Context-aware pervasive service composition and its implementation." In: *Personal and ubiquitous computing* 15.3, pp. 291–303. ISSN: 1617-4909. DOI: 10.1007/s00779-010-0333-5.

6
References

Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). "Towards a Better Understanding of Context and Context-Awareness." In: *Handheld and Ubiquitous Computing*. Ed. by H.-W. Gellersen. Vol. 1707. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 304–307. ISBN: 978-3-540-66550-2. DOI: 10.1007/3-540-48157-5_29.

Accorsi, R., Ullrich, M., and van der Aalst, W. M. P. (2012). "Aktuelles Schlagwort: Process Mining." In: *Informatik Spektrum* 35.5, pp. 354–359.

Adelman, L. (1991). "Experiments, Quasi-Experiments, and Case Studies: A Review of Empirical Methods for Evaluating Decision Support Systems." In: *IEEE Transactions on Systems, Man, and Cybernetics* 21.2, pp. 293–301.

Afflerbach, P., Kastner, G., Krause, F., and Röglinger, M. (2014). "The Business Value of Process Flexibility." In: *Business & Information Systems Engineering* 6.4, pp. 203–214. ISSN: 1867-0202. DOI: 10.1007/s12599-014-0333-5.

Agarwal, V., Chafle, G., Dasgupta, K., Karnik, N., Kumar, A., Mittal, S., and Srivastava, B. (2005). "Synthy: A system for end to end composition of web services." In: *Journal of Web Semantics* 3.4. World Wide Web Conference 2005——Semantic Web Track, pp. 311–339. ISSN: 1570-8268. DOI: 10.1016/j.websem.2005.09.002.

Aguilar-Savén, R. S. (2004). "Business process modelling: Review and framework." In: *International Journal of Production Economics* 90.2, pp. 129–149. ISSN: 09255273. DOI: 10.1016/S0925-5273(03)00102-6.

Alférez, G. H., Pelechano, V., Mazo, R., Salinesi, C., and Diaz, D. (May 2014). "Dynamic adaptation of service compositions with variability models." In: *Journal of Systems and Software* 91, pp. 24–47.

Armistead, C. and Machin, S. (Jan. 1997). "Implications of business process management for operations management." In: *International Journal of Operations & Production Management* 17.9, pp. 886–898. DOI: 10.1108/01443579710171217.

Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.

Attard, J., Scerri, S., Rivera, I., and Handschuh, S. (Sept. 2013). "Ontology-based situation recognition for context-aware systems." In: *Proceedings of the 9th International Conference on Semantic Systems*. ACM, pp. 113–120. ISBN: 1450319726. DOI: 10.1145/2506182.2506197.

Augusto, A., Conforti, R., Dumas, M., Rosa, M. L., Maggi, F. M., Marrella, A., Mecella, M., and Soo, A. (Apr. 2019). "Automated Discovery of Process Models from Event Logs: Review and Benchmark." In: *IEEE Transactions on Knowledge and Data Engineering* 31.4, pp. 686–705. ISSN: 2326-3865. DOI: 10.1109/TKDE.2018.2841877.

Ayora, C., Torres, V., Reichert, M., Weber, B., and Pelechano, V. (2013). "Towards Runtime Flexibility for Process Families: Open Issues and Research Challenges." In: *Busi-*

ness Process Management Workshops Lecture Notes in Business Information Processing 132, pp. 477–488.

Bach, M. P., Vukšić, V. B., Vugec, D. S., and Stjepić, A.-M. (Sept. 2019). "BPM and BI in SMEs: The role of BPM/BI alignment in organizational performance." In: *International Journal of Engineering Business Management* 11. DOI: 10.1177/1847979019874182.

Baldauf, M., Dustdar, S., and Rosenberg, F. (2007). "A survey on context-aware systems." In: *International Journal of Ad Hoc and Ubiquitous Computing* 2.4, pp. 263–277. ISSN: 1743-8225.

Bandara, W., Alibabaei, A., and Aghdasi, M. (Sept. 2009). "Means of achieving Business Process Management success factors." In: *Proceedings of the 4th Mediterranean Conference on Information Systems*. Ed. by P. Ein-Dor, A. Poulymenakou, and M. Amami. Athens University of Economics and Business, Athens, Greece: Department of Management Science & Technology, Athens University of Economics and Business, pp. 25–27. URL: http://eprints.qut.edu.au/30074/.

Bandara, W., Gable, G. G., and Rosemann, M. (2005). "Factors and measures of business process modelling: Model building through a multiple case study." In: *European Journal of Information Systems* 14.4, pp. 347–360. ISSN: 0960-085X. DOI: 10.1057/palgrave.ejis.3000546.

Bandara, W., Gable, G. G., and Rosemann, M. (2006). "Business process modeling success: An empirically tested measurement model." In: *Proceedings of the 27th International Conference on Information Systems*. Ed. by D. Straub and S. Klein. Milwaukee, Wisconsin: University of Wisconsin, pp. 1–20. URL: http://eprints.qut.edu.au/11131/.

Bandara, W., Indulska, M., Chong, S., and Sadiq, S. (2007). "Major issues in business process management: an expert perspective." In: *Proceedings of the 15th European Conference on Information Systems (ECIS)*, p. 89.

Barba, I., Del Valle, C., Weber, B., and Jiménez-Ramírez, A. (2013a). "Automatic generation of optimized business process models from constraint-based specifications." In: *International Journal of Cooperative Information Systems* 22.02, p. 1350009. DOI: 10.1142/S0218843013500093.

Barba, I., Weber, B., Del Valle, C., and Jiménez-Ramírez, A. (2013b). "User recommendations for the optimized execution of business processes." In: *Data & Knowledge Engineering* 86, pp. 61–84. DOI: 10.1016/j.datak.2013.01.004.

Bashari, M., Bagheri, E., and Du, W. (Dec. 2018). "Automated composition and optimization of services for variability-intensive domains." In: *Journal of Systems and Software* 146, pp. 356–376. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2018.07.039. URL: http://www.sciencedirect.com/science/article/pii/S0164121218301481.

Bauer, T. (2019). "Business Processes with Pre-modelled Flexibility: Examples and Requirements for Various Process Aspects." In: *American Journal of Management* 19.4.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., and Jeffries, R. (2001). *The Agile Manifesto*. URL: http://www.agilemanifesto.org.

Becker, J., Algermissen, L., Falk, T., Pfeiffer, D., and Fuchs, P. (July 2006). "Model Based Identification and Measurement of Reorganization Potential in Public Administrations – the PICTURE-Approach." In: *Proceedings of the 10th Pacific Asia Conference on Information Systems*. Kuala Lumpur, Malaysia, pp. 860–875. URL: http://www.pacis-net.org/file/2006/1148.pdf.

Becker, J., Beverungen, D., Knackstedt, R., Matzner, M., Müller, O., and Pöppelbuß, J. (2013). "Designing Interaction Routines in Service Networks." In: *Scandinavian Journal of Information Systems* 25.1, pp. 37–68.

Becker, J., Pfeiffer, D., Räckers, M., Falk, T., and Czerwonka, M. (2015). "Semantic Business Process Modelling and Analysis." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 187–217. ISBN: 978-3-642-45099-0. DOI: 10.1007/978-3-642-45100-3_9.

Becker, J., Rosemann, M., and von Uthmann, C. (2000). "Guidelines of Business Process Modeling." In: *Business Process Management*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, W. M. P. van der Aalst, J. Desel, and A. Oberweis. Vol. 1806. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 30–49. ISBN: 978-3-540-67454-2. DOI: 10.1007/3-540-45594-9_3.

Becker, J., Thome, I., Weiß, B., and Winkelmann, A. (July 2010a). "Constructing a Semantic Business Process Modelling Language for the Banking Sector: An Evolutionary Dyadic Design Science Approach." In: *Enterprise Modelling and Information Systems Architectures* 5.1, pp. 4–25. DOI: 10.18417/emisa.5.1.1. URL: https://www.emisa-journal.org/emisa/article/view/64.

Becker, J., Weiß, B., and Winkelmann, A. (2010b). "Utility vs. Efforts of Business Process Modeling — An Exploratory Survey in the Financial Sector." In: *Proceedings of the Multikonferenz Wirtschaftsinformatik 2010*. Ed. by M. Schumann, L. M. Kolbe, M. H. Breitner, and A. Frerichs. Göttingen: Universitätsverlag Göttingen, pp. 41–54. ISBN: 978-3-941875-31-9. URL: http://udoo.uni-muenster.de/downloads/publications/2361.pdf.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2001). "Planning in nondeterministic domains under partial observability via symbolic model checking." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)* 1, pp. 473–478.

Bertoli, P., Cimatti, A., Roveri, M., and Traverso, P. (2006). "Strong planning under partial observability." In: *Artificial Intelligence* 170.4–5, pp. 337–384. ISSN: 0004-3702. DOI: 10.1016/j.artint.2006.01.004. URL: http://www.sciencedirect.com/science/article/pii/S0004370206000075.

Bertoli, P., Pistore, M., and Traverso, P. (2010). "Automated composition of Web services via planning in asynchronous domains." In: *Artificial Intelligence* 174.3-4, pp. 316–361. ISSN: 0004-3702. DOI: 10.1016/j.artint.2009.12.002.

Bertrand, J. W. M. and Fransoo, J. C. (2002). "Operations management research methodologies using quantitative modeling." In: *International Journal of Operations & Production Management* 22.2, pp. 241–264.

Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., and Riboni, D. (2010). "A survey of context modelling and reasoning techniques." In: *Pervasive and mobile computing* 6.2, pp. 161–180. DOI: 10.1016/j.pmcj.2009.06.002. URL: http://www.sciencedirect.com/science/article/pii/S1574119209000510.

Betz, S., Klink, S., Koschmider, A., and Oberweis, A. (2006). "Automatic user support for business process modeling." In: *Proceedings of the 3rd European Semantic Web Conference (ESWC 2006)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.83.4899.

Bider, I. (2005). "Masking flexibility behind rigidity: Notes on how much flexibility people are willing to cope with." In: *Proceedings of the CAiSE* 5, pp. 7–18.

Bonet, B. and Geffner, H. E. (2001). "GPT: A Tool for Planning with Uncertainty and Partial Information." In: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pp. 82–87.

Bortlik, M., Heinrich, B., and Mayer, M. (Oct. 2018). "Multi User Context-Aware Service Selection for Mobile Environments." In: *Business & Information Systems Engineering* 60.5, pp. 415–430. DOI: 10.1007/s12599-018-0552-2.

Bose, R. J. C., van der Aalst, W. M. P., Zliobaite, I., and Pechenizkiy, M. (2014). "Dealing with concept drifts in process mining." In: *IEEE transactions on neural networks and learning systems* 25.1, pp. 154–171. ISSN: 2162-2388. DOI: 10.1109/TNNLS.2013.2278313.

Boukhedouma, S., Alimazighi, Z., and Oussalah, M. (2017). "Adaptation and Evolution Frameworks for Service Based Inter-Organizational Workflows." In: *International Journal of E-Business Research* 13.2, pp. 28–57. ISSN: 1548-1131. DOI: 10.4018/IJEBR.2017040103.

Bowers, J., Button, G., and Sharrock, W. (1995). "Workflow From Within and Without: Technology and Cooperative Work on the Print Industry Shopfloor." In: *Proceedings of the Fourth European Conference on Computer-Supported Cooperative Work (ECSCW '95)*. Ed. by H. Marmolin, Y. Sundblad, and K. Schmidt. Dordrecht: Springer Netherlands, pp. 51–66. ISBN: 978-94-011-0349-7. DOI: 10.1007/978-94-011-0349-7_4.

Branco, M. C., Xiong, Y., Czarnecki, K., Küster, J., and Völzer, H. (2014). "A case study on consistency management of business and IT process models in banking." In: *Software & Systems Modeling* 13.3, pp. 913–940. ISSN: 1619-1366. DOI: 10.1007/s10270-013-0318-8.

Braunnagel, D., Falk, T., Wehner, B., and Leist, S. (June 2016). "BPM Adoption in Small and Medium-sized Companies in Bavaria." In: *Proceedings of the 24th European Conference on Information Systems (ECIS)*. URL: https://epub.uni-regensburg.de/34064/.

Brockmans, S., Ehrig, M., Koschmider, A., Oberweis, A., and Studer, R. (2006). "Semantic Alignment Of Business Processes." In: *Proceedings of the Eighth International Conference on Enterprise Information Systems (ICEIS 2006)*, pp. 191–196.

Bucchiarone, A., Mezzina, C., and Pistore, M. (2013). "Captlang: a language for context-aware and adaptable business processes." In: *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*.

Bucchiarone, A., Pistore, M., Raik, H., and Kazhamiakin, R. (2011). "Adaptation of service-based business processes by context-aware replanning." In: *Service-Oriented Computing and Applications (SOCA), 2011 IEEE International Conference on*. IEEE, pp. 1–8. ISBN: 1467303186.

Business Process Model and Notation (BPMN): Version 2.0.2 (2013). URL: http://www.omg.org/spec/BPMN/2.0.2/PDF (visited on 10/16/2014).

Cabanillas, C., Resinas, M., Mendling, J., and Ruiz-Cortés, A. (2015). "Automated team selection and compliance checking in business processes." In: *Proceedings of the 2015 International Conference on Software and System Process*, pp. 42–51.

Canfora, G., Di Penta, M., Esposito, R., and Villani, M. L. (2005). "An Approach for QoS-aware Service Composition based on Genetic Algorithms." In: *Proceedings of the 7th annual conference on Genetic and evolutionary computation (GECCO '05)*, pp. 1069–1075.

Cardoso, J. (2007). "Complexity analysis of BPEL Web processes." In: *Software Process: Improvement and Practice* 12.1, pp. 35–49. ISSN: 10774866. DOI: 10.1002/spip.302.

Chafle, G., Dasgupta, K., Kumar, A., Mittal, S., and Srivastava, B. (2006). "Adaptation in Web Service Composition and Execution." In: *Proceedings of the 2006 IEEE International Conference on Web Services (ICWS'06)*, pp. 549–557.

Chen, H., Finin, T., and Joshi, A. (2003). "An Ontology for Context-Aware Pervasive Computing Environments." In: *The Knowledge Engineering Review* 18.3, pp. 197–207.

Chen, L., Li, X., and Yang, Q. (2012). "Continuous process improvement based on adaptive workflow mining technique." In: *Journal of Computational Information Systems* 8.7, pp. 2891–2898.

Chen, Q. and Hsu, M. (2001). "Inter-enterprise collaborative business process management." In: *Proceedings*. Los Alamitos, Calif. [u.a.]: IEEE Computer Society, pp. 253–260. ISBN: 0-7695-1001-9.

Chinosi, M. and Trombetta, A. (2012). "BPMN: An introduction to the standard." In: *Computer Standards & Interfaces* 34.1, pp. 124–134. ISSN: 0920-5489.

Chiu, D., Karlapalem, K., and Li, Q. (2003). "Views for Inter-organization Workflow in an E-commerce Environment." In: *Semantic Issues in E-Commerce Systems*. Ed. by R. Meersman, K. Aberer, and T. Dillon. Vol. 111. IFIP - The International Federation for Information Processing. Springer US, pp. 137–151. ISBN: 978-1-4757-1035-9. DOI: 10.1007/978-0-387-35658-7_9. URL: http://dx.doi.org/10.1007/978-0-387-35658-7_9.

Chouhan, S. S. and Niyogi, R. (2017). "MAPJA: Multi-agent planning with joint actions." In: *Applied Intelligence* 14.4, p. 105. DOI: 10.1007/s10489-017-0938-8.

Cimatti, A., Do, M., Micheli, A., Roveri, M., and Smith, D. E. (2018). "Strong temporal planning with uncontrollable durations." In: *Artificial Intelligence* 256, pp. 1–34. ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2017.11.006. URL: http://www.sciencedirect.com/science/article/pii/S0004370217301431.

Cobham, A. (1965). "The intrinsic computational difficulty of functions." In: *Logic, Methodology, and Philosophy of Science II*.

Cognini, R., Corradini, F., Gnesi, S., Polini, A., and Re, B. (Apr. 2018). "Business process flexibility - a systematic literature review with a software systems perspective." In: *Information Systems Frontiers* 20.2, pp. 343–371. ISSN: 1387-3326. DOI: 10.1007/s10796-016-9678-2.

Crosby, M., Rovatsos, M., and Petrick, R. P. A. (2013). "Automated Agent Decomposition for Classical Planning." In: *Proceedings of the 23rd International Conference on Automated Planning and Scheduling*.

Davenport, T. H. (1993). *Process innovation: Reengineering work through information technology*. Boston, Mass.: Harvard Business School Press. ISBN: 9780875843667.

Davenport, T. H. and Short, J. (1990). *Information technology and business process redesign*. Taylor & Francis US.

De Leoni, M. and Marrella, A. (2017). "Aligning Real Process Executions and Prescriptive Process Models through Automated Planning." In: *Expert Systems with Applications* 82, pp. 162–183. ISSN: 09574174. DOI: 10.1016/j.eswa.2017.03.047.

De Morais, R. M., Kazan, S., de Pádua, S. I. D., and Costa, A. L. (2014). "An analysis of BPM lifecycles: from a literature review to a framework proposal." In: *Business Process Mgmt Journal* 20.3, pp. 412–432. ISSN: 1463-7154. DOI: 10.1108/BPMJ-03-2013-0035.

De Weerdt, M. and Clement, B. (2009). "Introduction to planning in multiagent systems." In: *Multiagent and Grid Systems - Planning in multiagent systems* 5.4, pp. 345–355.

Dey, A. K. (2001). "Understanding and using context." In: *Personal and ubiquitous computing* 5.1, pp. 4–7. ISSN: 1617-4909.

Dimopoulos, Y. and Moraitis, P. (2006). "Multi-Agent Coordination and Cooperation through Classical Planning." In: *2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pp. 398–402. DOI: 10.1109/IAT.2006.90.

Ding, Z., Sun, Y., Liu, J., Pan, M., and Liu, J. (2015). "A genetic algorithm based approach to transactional and QoS-aware service selection." In: *Enterprise Information Systems*, pp. 1–20. ISSN: 1751-7575. DOI: 10.1080/17517575.2015.1048832.

Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). "A survey of autonomic communications." In: *ACM Transactions on Autonomous and Adaptive Systems* (*TAAS*) 1.2, pp. 223–259. ISSN: 1556-4665.

Döhring, M., Reijers, H. A., and Smirnov, S. (2014). "Configuration vs. adaptation for business process variant maintenance: An empirical study." In: *Information Systems* 39, pp. 108–133. ISSN: 0306-4379. DOI: 10.1016/j.is.2013.06.002.

Dong, X., Halevy, A., Madhavan, J., Nemes, E., and Zhang, J. (2004). "Similarity search for web services." In: *Proceedings of the Thirtieth International Conference on Very Large Data Bases*. Vol. 30. St. Louis, MO: Morgan Kaufmann, pp. 372–383. ISBN: 0-12-088469-0.

Doran, D. (2004). "Rethinking the supply chain: an automotive perspective." In: *Supply Chain Management: An International Journal* 9.1, pp. 102–109.

Ehrig, M., Koschmider, A., and Oberweis, A. (Feb. 2007). "Measuring Similarity between Semantic Business Process Models." In: *Proceedings of the 4th Asia-Pacific Conference on Conceptual Modelling* (*APCCM 2007*). Ed. by J. F. Roddick and A. Hinze. Vol. 67. Ballarat, Victoria, Australia, pp. 71–80.

Eisenbarth, T. (2013). *Semantic Process Models: Transformation, Adaptation, Resource Consideration*. Augsburg: Universität Augsburg.

Eisenbarth, T., Lautenbacher, F., and Bauer, B. (2011). "Adaptation of Process Models – A Semantic-based Approach." In: *Journal of Research and Practice in Information Technology* 43.1, pp. 5–23.

Ellis, C., Keddara, K., and Rozenberg, G. (1995). "Dynamic change within workflow systems." In: *Proceedings of the conference on Organizational computing system* (*COCS '95*). Ed. by N. Comstock and C. Ellis, pp. 10–21. DOI: 10.1145/224019.224021.

Ephrati, E. and Rosenschein, J. S. (Aug. 1994). "Divide and conquer in multi-agent planning." In: *Proceedings of the 12th National Conference on Artificial Intelligence*. Vol. 1. Seattle, WA, USA, p. 80.

Fahland, D., Favre, C., Koehler, J., Lohmann, N., Völzer, H., and Wolf, K. (2011). "Analysis on demand: Instantaneous soundness checking of industrial business process

models." In: *Data & Knowledge Engineering* 70.5, pp. 448–466. DOI: 10.1016/j.datak.2011.01.004.

Fahland, D. and van der Aalst, W. M. P. (2012). "Repairing Process Models to Reflect Reality." In: *Business Process Management* 7481, pp. 229–245.

Falou, M. E., Bouzid, M., Mouaddib, A.-I., and Vidal, T. (2009). "Automated Web Service Composition: A Decentralised Multi-agent Approach." In: *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pp. 387–394. DOI: 10.1109/WI-IAT.2009.68.

Fan, S.-L., Yang, Y.-B., and Wang, X.-X. (June 2018). "Efficient Web Service Composition via Knapsack-Variant Algorithm." In: *Proceedings of the 15th International Conference on Services Computing (SCC 2018)*. Ed. by J. E. Ferreira, G. Spanoudakis, Y. Ma, and L.-J. Zhang. Seattle, WA, USA: Springer International Publishing, pp. 51–66. ISBN: 978-3-319-94376-3.

Fellmann, M., Delfmann, P., Koschmider, A., Laue, R., Leopold, H., and Schoknecht, A. (2015a). "Semantic Technology in Business Process Modeling and Analysis: Part 1: Matching, Modeling Support, Correctness and Compliance." In: *Emisa Forum* 35.1.

Fellmann, M., Zarvic, N., Metzger, D., and Koschmider, A. (2015b). "Requirements Catalog for Business Process Modeling Recommender Systems." In: *Wirtschaftsinformatik Proceedings 2015*. Ed. by O. Thomas and F. Teuteberg, pp. 393–407.

Fikes, R. E. and Nilsson, N. J. (1971). "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving." In: *Proceedings of the 2Nd International Joint Conference on Artificial Intelligence*. IJCAI'71. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, pp. 608–620. URL: http://dl.acm.org/citation.cfm?id=1622876.1622939.

Fleischmann, A., Kannengiesser, U., Schmidt, W., and Stary, C. (2013). "Subject-Oriented Modeling and Execution of Multi-agent Business Processes." In: *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*. IEEE, pp. 138–145. ISBN: 978-0-7695-5145-6. DOI: 10.1109/WI-IAT.2013.102.

Forstner, E., Kamprath, N., and Röglinger, M. (2014). "Capability development with process maturity models – Decision framework and economic analysis." In: *Journal of Decision Systems* 23.2, pp. 127–150. ISSN: 1246-0125. DOI: 10.1080/12460125.2014.865310.

Frederiks, P. and van der Weide, T. (2006). "Information modeling: The process and the required competencies of its participants." In: *Data & Knowledge Engineering* 58.1, pp. 4–20. DOI: 10.1016/j.datak.2005.05.007.

Fujii, K. and Suda, T. (2009). "Semantics-based context-aware dynamic service composition." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 4.2, p. 12. ISSN: 1556-4665.

Gaaloul, W., Alaoui, S., Baïna, K., and Godart, C. (2005a). "Mining Workflow Patterns through Event-data Analysis." In: *Proceedings of the The 2005 Symposium on Applications and the Internet Workshops (SAINT-W'05)*, pp. 226–229.

Gaaloul, W., Baïna, K., and Godart, C. (2005b). "Towards Mining Structural Workflow Patterns." In: *Database and Expert Systems Applications*. Ed. by K. Andersen, J. Debenham, and R. Wagner. Vol. 3588. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 24–33. ISBN: 978-3-540-28566-3. DOI: 10.1007/11546924_3. URL: http://dx.doi.org/10.1007/11546924_3.

Gambini, M., La Rosa, M., Migliorini, S., and ter Hofstede, A. H. M. (2011). "Automated error correction of business process models." In: *Business Process Management*. Ed. by S. Rinderle-Ma, F. Tourmani, and K. Wolf. Springer, pp. 148–165.

García-Bañuelos, L., van Beest, N., Dumas, M., La Rosa, M., and Mertens, W. (Mar. 2018). "Complete and Interpretable Conformance Checking of Business Processes." In: *IEEE Transactions on Software Engineering* 44.3, pp. 262–290. ISSN: 2326-3881. DOI: 10.1109/TSE.2017.2668418.

Garrido, A., Guzman, C., and Onaindia, E. (2010). "Anytime plan-adaptation for continuous planning." In: *PlanSIG'10*, pp. 62–69.

Gerevini, A. and Serina, I. (Apr. 2000). "Fast Plan Adaptation through Planning Graphs: Local and Systematic Search Techniques." In: *Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems*. AIPS'00. Breckenridge, CO, USA: AAAI Press, pp. 112–121. ISBN: 1577351118.

Gerevini, A. E., Saetti, A., and Serina, I. (2012). "Case-based Planning for Problems with Real-valued Fluents: Kernel Functions for Effective Plan Retrieval." In: *Frontiers in Artificial Intelligence and Applications* 242.ECAI 2012, pp. 348–353.

Ghallab, M., Nau, D. S., and Traverso, P. (2004). *Automated Planning: Theory & Practice*. San Francisco: Morgan Kaufmann. ISBN: 0080490514.

Ghallab, M., Nau, D. S., and Traverso, P. (2016). *Automated Planning and Acting*. New York, NY: Cambridge University Press. ISBN: 978-1107037274.

Ghrab, S., Saad, I., Kassel, G., and Gargouri, F. (2017). "A Core Ontology of Know-How and Knowing-That for improving knowledge sharing and decision making in the digital age." In: *Journal of Decision Systems* 26.2, pp. 138–151. ISSN: 1246-0125.

Glavan, L. M. and Vukšić, V. B. (2017). "Examining the impact of business process orientation on organizational performance: the case of Croatia." In: *Croatian Operational Research Review* 8.1, pp. 137–165.

Goldman, S. L., Nagel, R. N., and Preiss, K. (1994). *Agile Competitors and Virtual Organisations: Strategies for Enriching the Customer*. Van Nostrand Reinhold.

Gordijn, J., Akkermans, H., and van Vliet, H. (2000). "Business Modelling Is Not Process Modelling." In: *Conceptual Modeling for E-Business and the Web*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, S. W. Liddle, H. C. Mayr, and B. Thalheim. Vol. 1921. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 40–51. ISBN: 978-3-540-41073-7. DOI: 10.1007/3-540-45394-6_5.

Gottschalk, F. and La Rosa, M. (2010). "Process Configuration." In: *Modern Business Process Automation*. Ed. by A. H. M. ter Hofstede, W. M. P. van der Aalst, M. Adams, and N. Russell. Berlin Heidelberg: Springer, pp. 459–487. ISBN: 978-3-642-03122-9. DOI: 10.1007/978-3-642-03121-2_18.

Gottschalk, F., van der Aalst, W. M. P., and Jansen-Vullers, M. H. (2007). "Configurable process models—a foundational approach." In: *Reference Modeling*. Springer, pp. 59–77. ISBN: 3790819654.

Grefen, P., Aberer, K., Hoffner, Y., and Ludwig, H. (2000). "CrossFlow: cross-organizational workflow management in dynamic virtual enterprises." In: *International Journal of Computer Systems Science & Engineering* 15.5, pp. 277–290.

Gschwind, T., Koehler, J., and Wong, J. (2008). "Applying Patterns during Business Process Modeling: Business Process Management." In: *Business Process Management*. Ed. by M. Dumas, M. Reichert, and M.-C. Shan. Vol. 5240. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 4–19. ISBN: 978-3-540-85757-0. DOI: 10.1007/978-3-540-85758-7_4.

Hallerbach, A., Bauer, T., and Reichert, M. (2008a). "Context-based configuration of process variants." In: *Proceeding of the 3rd International Workshop on Technologies for Context-Aware Business Process Management (TCoB 2008)*.

Hallerbach, A., Bauer, T., and Reichert, M. (June 2008b). "Managing Process Variants in the Process Lifecycle." In: *Proceedings of the 10th International Conference on Enterprise Information Systems*. Barcelona, Spain, pp. 154–161.

Hallerbach, A., Bauer, T., and Reichert, M. (2010). "Capturing variability in business process models: the Provop approach." In: *Journal of Software Maintenance and Evolution: Research and Practice* 22.6–7, pp. 519–546. ISSN: 1532-0618.

Hammer, M. (2015). "What is Business Process Management?" In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 3–16. ISBN: 978-3-642-45099-0. DOI: 10.1007/978-3-642-45100-3_1.

Harmon, P. (2019). *Business Process Change - A Business Process Management Guide for Managers and Process Professionals*. Fourth Edition. Morgan Kaufmann.

Harraf, A., Wanasika, I., Tate, K., and Talbott, K. (2015). "Organizational agility." In: *Journal of Applied Business Research* 31.2, p. 675.

Havur, G., Cabanillas, C., Mendling, J., and Polleres, A. (2015). "Automated Resource Allocation in Business Processes with Answer Set Programming." In: *11th International Workshop on Business Process Intelligence*.

Havur, G., Cabanillas, C., Mendling, J., and Polleres, A. (2016). "Resource Allocation with Dependencies in Business Process Management Systems." In: *Business Process Management Forum: BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings*. Ed. by M. La Rosa, P. Loos, and O. Pastor. Cham: Springer International Publishing, pp. 3–19. ISBN: 978-3-319-45468-9. DOI: 10.1007/978-3-319-45468-9_1.

Heinrich, B., Bewernik, M.-A., Henneberger, M., Krammer, A., and Lautenbacher, F. (2008). "SEMPA - A Semantic Business Process Management Approach for the Planning of Process Models." In: *Business & Information Systems Engineering* (*formerly Wirtschaftsinformatik*) 50.6, 445–460 (in German). URL: http://epub.uni-regensburg.de/23173/.

Heinrich, B., Bolsinger, M., and Bewernik, M.-A. (2009). "Automated planning of process models: the construction of exclusive choices." In: *Proceedings of the 30th International Conference on Information Systems* (*ICIS*). Ed. by H. Chen and S. A. Slaughter. Phoenix, Arizona and USA: Springer, pp. 1–18. URL: http://epub.uni-regensburg.de/23591/.

Heinrich, B., Klier, M., Lewerenz, L., and Zimmermann, S. (Mar. 2015a). "Quality of Service-aware Service Selection: A Novel Approach Considering Potential Service Failures and Non-Deterministic Service Values." In: *INFORMS Service Science, A Journal of the Institute for Operations Research and the Management Sciences* 7.1, pp. 48–69. DOI: 10.1287/serv.2015.0093.

Heinrich, B., Klier, M., and Zimmermann, S. (2012). "Automated Planning of Process Models –Towards a Semantic-based Approach." In: *Smolnik, S.; Teuteberg, F.; Thomas, O. (Ed.): Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications. Hershey: IGI Global*, pp. 169–194. URL: http://epub.uni-regensburg.de/23349/.

Heinrich, B., Klier, M., and Zimmermann, S. (2015b). "Automated planning of process models: Design of a novel approach to construct exclusive choices." In: *Decision Support Systems* 78, pp. 1–14. DOI: 10.1016/j.dss.2015.07.005.

Heinrich, B., Krause, F., and Schiller, A. (2019). "Automated planning of process models: The construction of parallel splits and synchronizations." In: *Decision Support Systems* 125, p. 113096. ISSN: 0167-9236. DOI: 10.1016/j.dss.2019.113096. URL: http://www.sciencedirect.com/science/article/pii/S0167923619301253.

Heinrich, B. and Mayer, M. (2018a). "Service selection in mobile environments: considering multiple users and context-awareness." In: *Journal of Decision Systems* 27.2, pp. 92–122. DOI: 10.1080/12460125.2018.1513223.

Heinrich, B., Schiller, A., and Schön, D. (2018b). "The cooperation of multiple actors within process models: an automated planning approach." In: *Journal of Decision Systems* 27.4, pp. 238–274. DOI: 10.1080/12460125.2019.1600894. URL: https://epub.uni-regensburg.de/40166/.

Heinrich, B. and Schön, D. (2015c). "Automated Planning of context-aware Process Models." In: *Proceedings of the 23rd European Conference on Information Systems (ECIS)*. Ed. by J. Becker, J. vom Brocke, and M. de Marco. Münster, Germany, Paper 75.

Heinrich, B. and Schön, D. (2016). "Automated Planning of Process Models: The Construction of Simple Merges." In: *Proceedings of the 24rd European Conference on Information Systems (ECIS)*. Istanbul, Turkey.

Henneberger, M., Heinrich, B., Lautenbacher, F., and Bauer, B. (2008). "Semantic-Based Planning of Process Models." In: *Multikonferenz Wirtschaftsinformatik (MKWI)*. Ed. by M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, A. Picot, B. Speitkamp, and P. Wolf. München: GITO-Verlag, pp. 1677–1689. URL: http://epub.uni-regensburg.de/23594/.

Hepp, M. and Dumitri, R. (2007). "An Ontology Framework for Semantic Business Process Management." In: *Proceedings of the 8th International Conference on Business Informatics (WI 2007): eOrganisation: Service-*, pp. 423–440.

Hepp, M., Leymann, F., Bussler, C., Domingue, J., Wahler, A., and Fensel, D. (2005). "Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management." In: *IEEE International Conference on E-Business Engineering (ICEBE 2005)* 0, pp. 535–540. DOI: 10.1109/ICEBE.2005.110.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). "Design science in information systems research." In: *MIS Q* 28.1, pp. 75–105. ISSN: 0276-7783.

Hoffmann, J. and Brafman, R. I. (2005). "Contingent Planning via Heuristic Forward Search with Implicit Belief States." In: *Proceedings of the 15th International Conference on Automated Planning and Scheduling*. ICAPS'05. AAAI Press, pp. 71–88. ISBN: 1-57735-220-3.

Hoffmann, J., Weber, I., and Kraft, F. M. (2009). "Planning@ sap: An application in business process management." In: *Proceedings of the 2nd International Scheduling and Planning Applications woRKshop (SPARK'09)*.

Hoffmann, J., Weber, I., and Kraft, F. M. (July 2012). "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management." In: *Journal of Artificial Intelligence Research* 44.1, pp. 587–632. DOI: 10.1613/jair.3636.

Hoppenbrouwers, S. J. B. A., Proper, H. A., and van der Weide, T. P. (2005). "A Fundamental View on the Process of Conceptual Modeling." In: *Conceptual Modeling – ER 2005*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, L. Delcambre, C. Kop, H. C. Mayr, J. Mylopoulos, and O. Pastor. Vol. 3716. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 128–143. ISBN: 978-3-540-29389-7. DOI: 10.1007/11568322_9.

Hornung, T., Koschmider, A., and Lausen, G. (2008). "Recommendation Based Process Modeling Support: Method and User Experience." In: *Conceptual Modeling - ER 2008*. Ed. by Q. Li, S. Spaccapietra, E. Yu, and A. Olivé. Vol. 5231. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 265–278. ISBN: 978-3-540-87876-6. DOI: 10.1007/978-3-540-87877-3_20.

Hornung, T., Koschmider, A., and Oberweis, A. (2007). "A Rule-based Autocompletion Of Business Process Models." In: *CAiSE Forum 2007. Proceedings of the 19th Conference on Advanced Information Systems Engineering (CAiSE)*. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.6389.

Hu, B., Wang, Z., and Dong, Q. (Oct. 2012). "A Modeling and Reasoning Approach Using Description Logic for Context-Aware Pervasive Computing: Emerging Research in Artificial Intelligence and Computational Intelligence." In: *Emerging Research in Artificial Intelligence and Computational Intelligence, Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence*. Ed. by J. Lei, F. L. Wang, H. Deng, and D. Miao. Vol. 315. Communications in Computer and Information Science. Springer Berlin Heidelberg, pp. 155–165. ISBN: 978-3-642-34240-0. DOI: 10.1007/978-3-642-34240-0_21.

Huang, J. S., Hsueh, K., and Reynolds, A. (2013). "A framework for collaborative social, economic and environmental development: Building a digital ecosystem for societal empowerment." In: *7th IEEE International Conference on Digital Ecosystems and Technologies (DEST) - Complex Environment Engineering*, pp. 166–171. DOI: 10.1109/DEST.2013.6611348.

Hull, R. and Motahari Nezhad, H. R. (2016). "Rethinking BPM in a Cognitive World: Transforming How We Learn and Perform Business Processes." In: *Business Process Management*. Ed. by M. La Rosa, P. Loos, and O. Pastor. Cham: Springer International Publishing, pp. 3–19. ISBN: 978-3-319-45348-4.

IEEE Task Force on Process Mining (2012). "Process Mining Manifesto." In: *Business Process Management Workshops*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, F. Daniel, K. Barkaoui, and S. Dustdar. Vol. 99. Berlin,

Heidelberg: Springer Berlin Heidelberg, pp. 169–194. ISBN: 978-3-642-28107-5. DOI: 10.1007/978-3-642-28108-2_19.

Indulska, M., Recker, J., Rosemann, M., and Green, P. (2009). "Business Process Modeling: Current Issues and Future Challenges." In: *Advanced Information Systems Engineering*. Ed. by P. van Eck, J. Gordijn, and R. Wieringa. Vol. 5565. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 501–514. ISBN: 978-3-642-02143-5. DOI: 10.1007/978-3-642-02144-2_39.

Jedlitschka, A., Ciolkowski, M., and Pfahl, D. (2008). "Reporting Experiments in Software Engineering." In: *Guide to advanced empirical software engineering*. Ed. by F. Shull, J. Singer, and D. I. K. Sjøberg. London: Springer, pp. 201–228. ISBN: 978-1-84800-044-5.

Jennings, N. R., Norman, T. J., Faratin, P., O'Brien, P., and Odgers, B. (2000). "Autonomous agents for business process management." In: *Applied Artificial Intelligence* 14.2, pp. 145–189. ISSN: 0883-9514. DOI: 10.1080/088395100117106.

Jiménez-Ramírez, A., Barba, I., Del Valle, C., and Weber, B. (2013). "Generating Multi-objective Optimized Business Process Enactment Plans." In: *Advanced Information Systems Engineering*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. Salinesi, M. C. Norrie, and Ó. Pastor. Vol. 7908. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 99–115. ISBN: 978-3-642-38708-1. DOI: 10.1007/978-3-642-38709-8_7.

Jin-Hai, L., Anderson, A. R., and Harrison, R. T. (2003). "The evolution of agile manufacturing." In: *Business Process Management Journal* 9.2, pp. 170–189.

Kalenkova, A. A., van der Aalst, W. M. P., Lomazova, I. A., and Rubin, V. A. (Oct. 2017). "Process Mining Using BPMN: Relating Event Logs and Process Models // Process mining using BPMN: Relating event logs and process models." In: *Software and Systems Modeling* 16.4, pp. 1019–1048. DOI: 10.1007/s10270-015-0502-0.

Kambhampati, S. (1997). "Refinement Planning as a Unifying Framework for Plan Synthesis." In: *AI MAGAZINE* 18.2, pp. 67–98.

Kannengiesser, U. (2017). "The Future: Obstacles and Opportunities." In: *S-BPM in the Production Industry: A Stakeholder Approach*. Ed. by M. Neubauer and C. Stary. Cham: Springer International Publishing, pp. 209–230. ISBN: 978-3-319-48466-2. DOI: 10.1007/978-3-319-48466-2_8.

Katzmarzik, A., Henneberger, M., and Buhl, H. U. (2012). "Interdependencies between automation and sourcing of business processes." In: *Journal of Decision Systems* 21.4, pp. 331–352. ISSN: 1246-0125. DOI: 10.1080/12460125.2012.749755.

Keller, G., Nüttgens, M., and Scheer, A.-W. (1992). "Semantische Prozeßmodellierung auf der Grundlage'Ereignisgesteuerter Prozeßketten (EPK)." In: *Veröffentlichungen des Instituts für Wirtschaftsinformatik*. Vol. 89. Saarbrücken: Universität Saarbrücken.

Képes, K., Breitenbücher, U., Gómez Sáez, S., Guth, J., Leymann, F., and Wieland, M. (2016). "Situation-Aware Execution and Dynamic Adaptation of Traditional Workflow Models." In: *Service-Oriented and Cloud Computing*. Ed. by M. Aiello, E. B. Johnsen, S. Dustdar, and I. Georgievski. Cham: Springer International Publishing, pp. 69–83. ISBN: 978-3-319-44482-6. DOI: 10.1007/978-3-319-44482-6_5.

Khan, F. H., Bashir, S., Javed, M. Y., Khan, A., and Khiyal, M. S. H. (2010). "QoS Based Dynamic Web Services Composition & Execution." In: *International Journal of Computer Science and Information Security (IJCSIS)* 7.2, pp. 147–152.

Kindler, E., Rubin, V., and Schäfer, W. (2006). "Process Mining and Petri Net Synthesis." In: *Business Process Management Workshops*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Eder, and S. Dustdar. Vol. 4103. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 105–116. ISBN: 978-3-540-38444-1. DOI: 10.1007/11837862_12.

Kitchenham, B. A., Pfleeger, S. L., Pickard, L. M., Jones, P. W., Hoaglin, D. C., El Emam, K., and Rosenberg, J. (2002). "Preliminary guidelines for empirical research in software engineering." In: *Software Engineering, IEEE Transactions on* 28.8, pp. 721–734. ISSN: 0098-5589. DOI: 10.1109/TSE.2002.1027796.

Klier, J., Klier, M., Müller, A.-L., and Rauch, C. (2016). "The impact of self-service technologies – towards an economic decision model and its application at the German Federal Employment Agency." In: *Journal of Decision Systems* 25.2, pp. 151–172. ISSN: 1246-0125. DOI: 10.1080/12460125.2016.1141274.

Koschmider, A. (2007). *Ähnlichkeitsbasierte Modellierungsunterstützung für Geschäftsprozesse*. Karlsruhe: Univ.-Verl. Karlsruhe. ISBN: 978-3-86644-188-0.

Koschmider, A., Hornung, T., and Oberweis, A. (2011). "Recommendation-based editor for business process modeling." In: *Data & Knowledge Engineering* 70.6, pp. 483–503. DOI: 10.1016/j.datak.2011.02.002.

Kossak, F., Illibauer, C., Geist, V., Natschläger, C., Ziebermayr, T., Freudenthaler, B., Kopetzky, T., and Schewe, K.-D. (2016). "A Layered Approach for Actor Modelling." In: *Hagenberg Business Process Modelling Method*. Ed. by F. Kossak, C. Illibauer, V. Geist, C. Natschläger, T. Ziebermayr, B. Freudenthaler, T. Kopetzky, and K.-D. Schewe. Cham: Springer International Publishing, pp. 63–84. ISBN: 978-3-319-30495-3. DOI: 10.1007/978-3-319-30496-0_3.

Krause, F., Bewernik, M.-A., and Fridgen, G. (2013). "Valuation of Manual and Automated Process Redesign from a Business Perspective." In: *Business Process Management Journal* 19.1.

Krogstie, J., Sindre, G., and Jørgensen, H. (2006). "Process models representing knowledge for action: A revised quality framework." In: *European Journal of Information Systems* 15.1, pp. 91–102. ISSN: 0960-085X. DOI: 10.1057/palgrave.ejis.3000598.

Kusiak, A., Nick Larson, T., and Wang, J. (1994). "Reengineering of design and manufacturing processes." In: *Computers & Industrial Engineering* 26.3, pp. 521–536. ISSN: 03608352. DOI: 10.1016/0360-8352(94)90048-5.

La Rosa, M., Dumas, M., ter Hofstede, A. H. M., and Mendling, J. (2011a). "Configurable multi-perspective business process models." In: *Information Systems* 36.2, pp. 313–340. ISSN: 0306-4379.

La Rosa, M., Reijers, H. A., van der Aalst, W. M. P., Dijkman, R. M., Mendling, J., Dumas, M., and García-Bañuelos, L. (2011b). "APROMORE: An advanced process model repository." In: *Expert Systems with Applications* 38.6, pp. 7029–7040. ISSN: 09574174. DOI: 10.1016/j.eswa.2010.12.012.

La Rosa, M., ter Hofstede, A. H. M., Wohed, P., Reijers, H. A., Mendling, J., and van der Aalst, W. M. P. (2011c). "Managing Process Model Complexity via Concrete Syntax Modifications." In: *IEEE Transactions on Industrial Informatics* 7.2, pp. 255–265. DOI: 10.1109/TII.2011.2124467.

La Rosa, M., van der Aalst, W. M. P., Dumas, M., and Milani, F. P. (Mar. 2017). "Business Process Variability Modeling: A Survey." In: *ACM Comput. Surv.* 50.1. ISSN: 0360-0300. DOI: 10.1145/3041957.

Lambert, D. M., Emmelhainz, M. A., and Gardner, J. T. (1996). "Developing and Implementing Supply Chain Partnerships." In: *The International Journal of Logistics Management* 7.2, pp. 1–17.

Laue, R. and Mendling, J. (2010). "Structuredness and its significance for correctness of process models." In: *Information Systems and e-Business Management* 8.3, pp. 287–307. ISSN: 1617-9846. DOI: 10.1007/s10257-009-0120-x.

Lautenbacher, F., Eisenbarth, T., and Bauer, B. (2009). "Process model adaptation using semantic technologies." In: *2009 13th Enterprise Distributed Object Computing Conference Workshops, EDOCW*, pp. 301–309. DOI: 10.1109/EDOCW.2009.5331985.

Le Clair, C. (2013). *Make Business Agility A Key Corporate Attribute – It Could Be What Saves You*. URL: http://blogs.forrester.com/craig_le_clair/13-09-09-make_business_agility_a_key_corporate_attribute_it_could_be_what_saves_you.

Leemans, S. J. J., Fahland, D., and van der Aalst, W. M. P. (May 2018). "Scalable process discovery and conformance checking." In: *Software & Systems Modeling* 17.2, pp. 599–631. ISSN: 1619-1374. DOI: 10.1007/s10270-016-0545-x.

Leopold, H., Mendling, J., and Gunther, O. (2016). "Learning from Quality Issues of BPMN Models from Industry." In: *IEEE Software* 33.4, pp. 26–33. ISSN: 0740-7459. DOI: 10.1109/MS.2015.81.

Lerner, B. S., Christov, S., Osterweil, L. J., Bendraou, R., Kannengiesser, U., and Wise, A. (2010). "Exception handling patterns for process modeling." In: *Software Engineering, IEEE Transactions on* 36.2, pp. 162–183. ISSN: 0098-5589.

Lewerenz, L. (2015). "A Heuristic Technique for an Efficient Decision Support in Context-aware Service Selection." In: *Proceedings of the 36th International Conference on Information Systems (ICIS)*, pp. 1–20. URL: https://epub.uni-regensburg.de/32660/.

Leymann, F., Roller, D., and Schmidt, M. T. (2002). "Web services and business process management." In: *IBM Systems Journal* 41.2, pp. 198–211.

Lin, S.-Y., Lin, G.-T., Chao, K.-M., and Lo, C.-C. (2012). "A Cost-Effective Planning Graph Approach for Large-Scale Web Service Composition." In: *Mathematical Problems in Engineering* 2012.1, pp. 1–21. ISSN: 1024-123X. DOI: 10.1155/2012/783476.

Linden, I., Derbali, M., Schwanen, G., Jacquet, J.-M., Ramdoyal, R., and Ponsard, C. (2014). "Supporting Business Process Exception Management by Dynamically Building Processes Using the BEM Framework." In: *Decision Support Systems III - Impact of Decision Support Systems for Global Environments*. Ed. by F. Dargam, J. E. Hernández, P. Zaraté, S. Liu, R. Ribeiro, B. Delibašić, and J. Papathanasiou. Vol. 184. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, pp. 67–78. ISBN: 978-3-319-11363-0. DOI: 10.1007/978-3-319-11364-7_7.

Lindland, O. I., Sindre, G., and Solvberg, A. (1994). "Understanding quality in conceptual modeling." In: *IEEE Software* 11.2, pp. 42–49. ISSN: 0740-7459. DOI: 10.1109/52.268955.

Liu, C., Zeng, Q., Duan, H., and Lu, F. (2015). "Petri Net Based Behavior Description of Cross-Organization Workflow with Synchronous Interaction Pattern." In: *Process-Aware Systems*. Ed. by J. Cao, L. Wen, and X. Liu. Vol. 495. Communications in Computer and Information Science. Springer Berlin Heidelberg, pp. 1–10. ISBN: 978-3-662-46169-3. DOI: 10.1007/978-3-662-46170-9_1.

Liu, C. and Zhang, F. (2016). "Petri Net Based Modeling and Correctness Verification of Collaborative Emergency Response Processes." In: *Cybernetics and Information Technologies* 16.3. ISSN: 1314-4081. DOI: 10.1515/cait-2016-0038.

Lusk, S., Paley, S., and Spanyi, A. (June 2005). "The Evolution of Business Process Management as a Professional Discipline." In: *BPTrends*. Association of Business Process Management Professionals International.

Mans, R., Reijers, H. A., Berends, H., Bandara, W., and Prince, R. (2013). "Business Process Mining Success." In: *Proceedings of the 21st European Conference on Information Systems*.

Marrella, A. (June 2019). "Automated Planning for Business Process Management." In: *Journal on Data Semantics* 8.2, pp. 79–98. ISSN: 1861-2040. DOI: 10.1007/s13740-018-0096-0.

Marrella, A. and Mecella, M. (2011a). "Continuous Planning for Solving Business Process Adaptivity." In: *Enterprise, Business-Process and Information Systems Modeling*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, T. Halpin, S. Nurcan, J. Krogstie, P. Soffer, E. Proper, R. Schmidt, and I. Bider. Vol. 81. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 118–132. ISBN: 978-3-642-21758-6. DOI: 10.1007/978-3-642-21759-3_9.

Marrella, A., Mecella, M., and Russo, A. (2011b). "Featuring Automatic Adaptivity through Workflow Enactment and Planning." In: *Proceedings of the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Ed. by D. Georgakopoulos and J. Joshi. DOI: 10.4108/icst.collaboratecom.2011.247096.

Marrella, A., Mecella, M., and Sardina, S. (2017). "Intelligent Process Adaptation in the SmartPM System." In: *ACM Transactions on Intelligent Systems and Technology* 8.2, pp. 1–43. ISSN: 21576904. DOI: 10.1145/2948071.

Marrella, A., Russo, A., and Mecella, M. (2012). "Planlets: Automatically Recovering Dynamic Processes in YAWL." In: *On the Move to Meaningful Internet Systems: OTM 2012*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, H. Panetto, T. Dillon, S. Rinderle-Ma, P. Dadam, X. Zhou, S. Pearson, A. Ferscha, S. Bergamaschi, and I. F. Cruz. Vol. 7565. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 268–286. ISBN: 978-3-642-33605-8. DOI: 10.1007/978-3-642-33606-5_17.

Martin, J. (1983). *Managing the data-base environment*. Englewood Cliffs, NJ: Prentice-Hall. ISBN: 9780135505823.

Masellis, R. D., Di Francescomarino, C., Ghidini, C., Montali, M., and Tessaris, S. (Feb. 2017). "Add Data into Business Process Verification: Bridging the Gap between Theory and Practice." In: *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. Ed. by S. Singh and S. Markovitch. AAAI'17. San Francisco, California, USA: AAAI Press, pp. 1091–1099. DOI: 10.5555/3298239.3298400.

Mathiesen, P., Bandara, W., Delavari, H., Harmon, P., and Brennan, K. (2011). "A comparative analysis of business analysis (BA) and business process management (BPM) capabilities." In: *Proceedings of the 19th European Conference on Information Systems (ECIS)*. Helsinki, Finland, p. 26.

Mattos, T. d. C., Santoro, F. M., Revoredo, K., and Nunes, V. T. (2014). "A formal representation for context-aware business processes." In: *Computers in Industry* 65.8, pp. 1193–1214. ISSN: 0166-3615. DOI: 10.1016/j.compind.2014.07.005. URL: http://www.sciencedirect.com/science/article/pii/S0166361514001407.

McElheran, K. (2015). "Do Market Leaders Lead in Business Process Innovation? The Case(s) of E-business Adoption." In: *Management Science* 61.6, pp. 1197–1216. ISSN: 0025-1909. DOI: 10.1287/mnsc.2014.2020.

Mehandjiev, N., Lécué, F., Carpenter, M., and Rabhi, F. A. (2012). "Cooperative Service Composition." In: *Advanced Information Systems Engineering*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Ralyté, X. Franch, S. Brinkkemper, and S. Wrycza. Vol. 7328. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 111–126. ISBN: 978-3-642-31094-2. DOI: 10.1007/978-3-642-31095-9_8.

Mei-hong, S., Shou-shan, J., Yong-gang, G., Liang, C., and Kai-duan, C. (2012). "A Method of Adaptive Process Mining Based on Time-Varying Sliding Window and Relation of Adjacent Event Dependency." In: *Intelligent System Design and Engineering Application (ISDEA), 2012 Second International Conference on*, pp. 24–31.

Mejri, A., Ayachi-Ghannouchi, S., and Martinho, R. (2020/02/08 2018). "A quantitative approach for measuring the degree of flexibility of business process models." In: *Business Process Management Journal (BPMJ)* 24.4, pp. 1023–1049. DOI: 10.1108/BPMJ-03-2017-0058.

Mendling, J., Neumann, G., and van der Aalst, W. M. P. (2007a). "Understanding the Occurrence of Errors in Process Models Based on Metrics." In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, and Z. Tari. Vol. 4803. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 113–130. ISBN: 978-3-540-76846-3. DOI: 10.1007/978-3-540-76848-7_9.

Mendling, J., Reijers, H. A., and Cardoso, J. (2007b). "What Makes Process Models Understandable?" In: *Business Process Management*. Ed. by G. Alonso, P. Dadam, and M. Rosemann. Vol. 4714. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–63. ISBN: 978-3-540-75182-3. DOI: 10.1007/978-3-540-75183-0_4.

Mendling, J., Reijers, H. A., and van der Aalst, W. M. P. (2010). "Seven process modeling guidelines (7PMG)." In: *Information and Software Technology* 52.2, pp. 127–136. ISSN: 0950-5849.

Mendling, J., Sánchez-González, L., García, F., and La Rosa, M. (2012a). "Thresholds for error probability measures of business process models." In: *Journal of Systems and Software* 85.5, pp. 1188–1197. DOI: 10.1016/j.jss.2012.01.017.

Mendling, J., Strembeck, M., and Recker, J. (2012b). "Factors of process model comprehension—Findings from a series of experiments." In: *Decision Support Systems* 53.1, pp. 195–206. DOI: 10.1016/j.dss.2011.12.013.

Mendling, J., Verbeek, H. M. W., van Dongen, B. F., van der Aalst, W. M. P., and Neumann, G. (2008). "Detection and prediction of errors in EPCs of the SAP reference model." In: *Data & Knowledge Engineering* 64.1, pp. 312–329. DOI: 10.1016/j.datak.2007.06.019.

Meredith, J. R., Raturi, A., Amoako-Gyampah, K., and Kaplan, B. (1989). "Alternative research paradigms in operations." In: *Journal of operations management* 8.4, pp. 297–326. ISSN: 0272-6963.

Meyer, H. and Weske, M. (2006). "Automated service composition using heuristic search." In: *Business Process Management*, pp. 81–96.

Milionis, N., Jereb, S., Henderson, K., Vrabic, J., Bain, M., Dolezal, J., Roessing, E., Dos Santos, J. N. C., Simeonova, R., and Otto, J. (Feb. 2019). *The EU's response to the "dieselgate" scandal*. URL: https://www.eca.europa.eu/lists/ecadocuments/brp_vehicle_emissions/brp_vehicle_emissions_en.pdf.

Minor, M., Tartakovski, A., and Bergmann, R. (2007). "Representation and Structure-Based Similarity Assessment for Agile Workflows." In: *Case-Based Reasoning Research and Development*. Ed. by R. O. Weber and M. M. Richter. Vol. 4626. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 224–238. ISBN: 978-3-540-74138-1. DOI: 10.1007/978-3-540-74141-1_16.

Mitroff, I. I., Betz, F., Pondy, L. R., and Sagasti, F. (1974). "On Managing Science in the Systems Age: Two Schemas for the Study of Science as a Whole Systems Phenomenon." In: *Interfaces* 4.3, pp. 46–58. ISSN: 0092-2102. DOI: 10.1287/inte.4.3.46.

Mitroff, I. I. and Mason, R. O. (1982). "Business Policy and Metaphysics: Some Philosophical Considerations." In: *Academy of Management Review* 7.3, pp. 361–371. DOI: 10.5465/amr.1982.4285320.

Montani, S., Leonardi, G., Quaglini, S., Cavallini, A., and Micieli, G. (2015). "A knowledge-intensive approach to process similarity calculation." In: *Expert Systems with Applications* 42.9, pp. 4207–4215. ISSN: 09574174. DOI: 10.1016/j.eswa.2015.01.027.

Montarnal, A., Mu, W., Benaben, F., Lamothe, J., Lauras, M., and Salatge, N. (2018). "Automated deduction of cross-organizational collaborative business processes." In: *Information Sciences* 453, pp. 30–49. ISSN: 0020-0255. DOI: 10.1016/j.ins.2018.03.041.

Moreno-Montes de Oca, I., Snoeck, M., Reijers, H. A., and Rodríguez-Morffi, A. (2015). "A systematic literature review of studies on business process modeling quality." In: *Information and Software Technology* 58, pp. 187–205. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2014.07.011.

Mulholland, A., Thomas, C. S., Kurchina, P., and Woods, D. (2006). *Mashup corporations: The end of business as usual*. Evolved Technologist Press.

Munoz-Gama, J., Carmona, J., and van der Aalst, W. M. (2014). "Single-Entry Single-Exit decomposed conformance checking." In: *Information Systems* 46, pp. 102–122. ISSN: 0306-4379. DOI: 10.1016/j.is.2014.04.003.

Nadoveza, D. and Kiritsis, D. (2014). "Ontology-based approach for context modeling in enterprise applications." In: *Special Issue on The Role of Ontologies in Future Web-based Industrial Enterprises* 65.9, pp. 1218–1231. ISSN: 0166-3615. DOI: 10.1016/j.compind.2014.07.007. URL: http://www.sciencedirect.com/science/article/pii/S0166361514001420.

Natschläger, C. and Geist, V. (2013). "A layered approach for actor modelling in business processes." In: *Business Process Management Journal* 19.6, pp. 917–932. ISSN: 1463-7154. DOI: 10.1108/BPMJ-10-2012-0107.

Nebel, B. and Koehler, J. (1995). "Plan reuse versus plan generation: A theoretical and empirical analysis." In: *Artificial Intelligence* 76.1-2, pp. 427–454. ISSN: 0004-3702. DOI: 10.1016/0004-3702(94)00082-C.

Netjes, M., Reijers, H. A., and van der Aalst, W. M. P. (2006). "Supporting the BPM lifecycle with FileNet." In: *Proceedings of the EMMSAD Workshop at the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06)*. Vol. 6, pp. 497–508.

Nissim, R., Brafman, R. I., and Domshlak, C. (2010). "A general, fully distributed multi-agent planning algorithm." In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*. Ed. by M. Luck. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, pp. 1323–1330. ISBN: 978-0-9826571-1-9.

Nunes, V. T., Santoro, F. M., Werner, C. M. L., and G. Ralha, C. (2018). "Real-Time Process Adaptation: A Context-Aware Replanning Approach." In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.1, pp. 99–118. ISSN: 2168-2216. DOI: 10.1109/TSMC.2016.2591538.

Nüttgens, M. and Rump, F. J. (2002). "Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK)." In: *Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen*. Ed. by J. Desel and M. Weske. LN in Informatics. GI-Edition.

Nüttgens, M. and Zimmermann, V. (1998). "Geschäftsprozeßmodellierung mit der objektorientierten Ereignisgesteuerten Prozeßkette (oEPK)." In: *Informationsmodellierung: Referenzmodelle und Werkzeuge*. Ed. by M. Maicher and H.-J. Scheruhn. Wiesbaden: Deutscher Universitätsverlag, pp. 23–35. ISBN: 978-3-663-07676-6. DOI: 10.1007/978-3-663-07676-6_2. URL: https://doi.org/10.1007/978-3-663-07676-6_2.

OMG Unified Modeling Language TM (OMG UML): Version 2.5 (2015). URL: http://www.omg.org/spec/UML/2.5 (visited on 05/04/2016).

Overhage, S., Birkmeier, D. Q., and Schlauderer, S. (2012). "Quality Marks, Metrics, and Measurement Procedures for Business Process Models." In: *Business & Information Systems Engineering* 4.5, pp. 229–246. ISSN: 1867-0202. DOI: 10.1007/s12599-012-0230-8.

Öztürk, P. and Aamodt, A. (1997). "Towards a Model of Context for Case-Based Diagnostic Problem Solving." In: *IN CONTEXT-97; PROCEEDINGS OF THE*, pp. 198–208.

Paik, I., Chen, W., and Huhns, M. N. (2014). "A scalable architecture for automatic service composition." In: *Services Computing, IEEE Transactions on* 7.1, pp. 82–95.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2008). "A Design Science Research Methodology for Information Systems Research." In: *Journal of Management Information Systems* 24.3, pp. 45–78.

Peleteiro, A., Burguillo, J. C., Arcos, J. L., and Rodriguez-Aguilar, J. A. (2014). "Fostering Cooperation Through Dynamic Coalition Formation and Partner Switching." In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 9.1, 1:1–1:31. ISSN: 1556-4665. DOI: 10.1145/2567928. URL: http://doi.acm.org/10.1145/2567928.

Pesic, M. (2008). "Constraint-based workflow management systems : shifting control to users." English. Proefschrift. PhD thesis. Department of Industrial Engineering Innovation Sciences. ISBN: 978-90-386-1319-2. DOI: 10.6100/IR638413.

Pesic, M., Schonenberg, M. H., Sidorova, N., and van der Aalst, Wil M. P. (2007). "Constraint-Based Workflow Models: Change Made Easy." In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, R. Meersman, and Z. Tari. Vol. 4803. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 77–94. ISBN: 978-3-540-76846-3. DOI: 10.1007/978-3-540-76848-7_7.

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

Pichler, P., Weber, B., Zugal, S., Pinggera, J., Mendling, J., and Reijers, H. (Aug. 2011). "Imperative versus Declarative Process Modeling Languages: An Empirical Investigation." In: *BPM 2011 Workshops, Part I, LNBIP* 99, pp. 383–394. DOI: 10.1007/978-3-642-28108-2_37.

Polyvyanyy, A., Vanhatalo, J., and Völzer, H. (2011). "Simplified Computation and Generalization of the Refined Process Structure Tree." In: *Proceedings of the 7th International Conference on Web Services and Formal Methods*. WS-FM'10. Berlin, Heidelberg:

Springer-Verlag, pp. 25–41. ISBN: 978-3-642-19588-4. URL: http://dl.acm.org/citation.cfm?id=1987781.1987783.

Pourmirza, S., Peters, S., Dijkman, R., and Grefen, P. (Feb. 2019). "BPMS-RA: A Novel Reference Architecture for Business Process Management Systems." In: *ACM Trans. Internet Technol.* 19.1. ISSN: 1533-5399. DOI: 10.1145/3232677.

Powell, S. G., Schwaninger, M., and Trimble, C. (2001). "Measurement and control of business processes." In: *System Dynamics Review* 17.1, pp. 63–91. ISSN: 0883-7066. DOI: 10.1002/sdr.206.

Prat, N., Comyn-Wattiau, I., and Akoka, J. (2015). "A Taxonomy of Evaluation Methods for Information Systems Artifacts." In: *Journal of Management Information Systems* 32.3, pp. 229–267. DOI: 10.1080/07421222.2015.1099390.

Prescher, J., Di Ciccio, C., and Mendling, J. (Nov. 2014). "From Declarative Processes to Imperative Models." In: *Proceedings of the Fourth International Symposium on Data-driven Process Discovery and Analysis* (*SIMPDA 2014*). Vol. 1293. Milan, Italy. DOI: 10.13140/2.1.1577.4409.

Pulgar, J. and Bastarrica, M. C. (2017). "Transforming Multi-role Activities in Software Processes into Business Processes." In: *Business Process Management Workshops: BPM 2016 International Workshops, Rio de Janeiro, Brazil, September 19, 2016, Revised Papers*. Ed. by M. Dumas and M. Fantinato. Cham: Springer International Publishing, pp. 372–383. ISBN: 978-3-319-58457-7. DOI: 10.1007/978-3-319-58457-7_27. URL: http://dx.doi.org/10.1007/978-3-319-58457-7_27.

Rao, J. and Su, X. (2005). "A Survey of Automated Web Service Composition Methods." In: *Semantic Web Services and Web Process Composition*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Cardoso, and A. Sheth. Vol. 3387. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 43–54. ISBN: 978-3-540-24328-1. DOI: 10.1007/978-3-540-30581-1_5.

Recker, J., Safrudin, N., and Rosemann, M. (2012). "How novices design business processes." In: *Information Systems* 37.6, pp. 557–573. ISSN: 0306-4379. DOI: 10.1016/j.is.2011.07.001.

Recker, J. C., Indulska, M., Rosemann, M., and Green, P. (2006). "How Good is BPMN Really? Insights from Theory and Practice." In: *Proceedings of the 14th European Conference on Information Systems* (*ECIS 2006*). Goeteborg, Sweden, Paper 135.

Regev, G., Bider, I., and Wegmann, A. (2007). "Defining business process flexibility with the help of invariants." In: *Software Process: Improvement and Practice* 12.1, pp. 65–79.

Reichert, M. and Dadam, P. (1997). "A framework for dynamic changes in workflow management systems." In: *Database and Expert Systems Applications, 1997. Proceedings., Eighth International Workshop on*, pp. 42–48.

Reichert, M. and Dadam, P. (1998). "Adeptflex—Supporting Dynamic Changes of Workflows Without Losing Control." In: *Journal of Intelligent Information Systems* 10.2, pp. 93–129. ISSN: 09259902. DOI: 10.1023/A:1008604709862.

Reichert, M. U. and Weber, B. (2012). *Enabling flexibility in process-aware information systems: challenges, methods, technologies*. Springer. ISBN: 3642304095.

Reisert, C., Zelt, S., and Wacker, J. (2018). "How to Move from Paper to Impact in Business Process Management: The Journey of SAP." In: *Business Process Management Cases: Digital Innovation and Business Transformation in Practice*. Ed. by J. vom Brocke and J. Mendling. Cham: Springer International Publishing, pp. 21–36. ISBN: 978-3-319-58307-5. DOI: 10.1007/978-3-319-58307-5_2.

Rinderle, S., Reichert, M., and Dadam, P. (2004). "Correctness criteria for dynamic changes in workflow systems—-a survey." In: *Data & Knowledge Engineering* 50.1, pp. 9–34. DOI: 10.1016/j.datak.2004.01.002.

Rosemann, M. (2006). "Potential pitfalls of process modeling: part A." In: *Business Process Management Journal* 12.2, pp. 249–254. DOI: 10.1108/14637150610657567. URL: http://www.ingentaconnect.com/content/mcb/157/2006/00000012/00000002/art00009.

Rosemann, M. and Bruin, T. D. (2005). "Towards a Business Process Management Maturity Model." In: *Proceedings of the Thirteenth European Conference on Information Systems (ECIS 2005)*. Ed. by F. Rajola, D. Avison, R. Winter, J. Becker, P. Ein-Dor, D. Bartmann, F. Bodendorf, C. Weinhardt, and J. Kallinikos. Verlag and the London School of Economics, pp. 1–12. URL: https://eprints.qut.edu.au/25194/.

Rosemann, M., Recker, J. C., and Flender, C. (2008). "Contextualisation of business processes." In: *International Journal of Business Process Integration and Management* 3.1, pp. 47–60. ISSN: 1741-8771.

Rosemann, M., Recker, J. C., and Flender, C. (2010). "Designing context-aware business processes." In: *Systems Analysis and Design: People, Processes, and Projects. Armonk, NY: ME Sharpe, Inc*, pp. 53–74.

Rosemann, M., Sedera, W., and Gable, G. G. (2001). "Critical success factors of process modeling for enterprise systems." In: *7th Americas Conference on Information Systems*. Boston, MA: Association for Information Systems, pp. 1128–1130. URL: http://eprints.qut.edu.au/25119/.

Rosemann, M. and van der Aalst, W. M. P. (2007). "A configurable reference modelling language." In: *Information Systems* 32.1, pp. 1–23. ISSN: 0306-4379.

Rosemann, M. and vom Brocke, J. (2015). "The Six Core Elements of Business Process Management." In: *Handbook on Business Process Management 1*. Ed. by J. vom Brocke and M. Rosemann. Heidelberg Dordrecht London New York: Springer, pp. 107–122.

Roy, S., Sajeev, A. S. M., Bihary, S., and Ranjan, A. (2014). "An Empirical Study of Error Patterns in Industrial Business Process Models." In: *IEEE Transactions on Services Computing* 7.2, pp. 140–153. ISSN: 1939-1374. DOI: 10.1109/TSC.2013.10.

Rozinat, A., Zickler, S., Veloso, M., van der Aalst, W. M. P., and McMillen, C. (2009). "Analyzing Multi-agent Activity Logs Using Process Mining Techniques." In: *Distributed Autonomous Robotic Systems 8*. Ed. by H. Asama, H. Kurokawa, J. Ota, and K. Sekiyama. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 251–260. ISBN: 978-3-642-00643-2. DOI: 10.1007/978-3-642-00644-9_22.

Russell, N., ter Hofstede, A. H. M., and Mulyar, N. (2006a). "Workflow ControlFlow Patterns: A Revised View." In: *BPM Center Report BPM-06-22*. URL: http://bpmcenter.org/reports.

Russell, N., van der Aalst, W. M. P., and Ter Hofstede, A. (2016). *Workflow patterns: The definitive guide*. Information systems. Cambridge, Massachusetts: The MIT Press. ISBN: 978-0-262-02982-7.

Russell, N., van der Aalst, W. M. P., and ter Hofstede, A. H. M. (2006b). "Exception Handling Patterns." In: *Process-Aware Information Systems. Technical report, BPM Center Report BPM-06-04, BPMcenter. org*.

Russell, N., van der Aalst, W. M. P., ter Hofstede, A. H. M., and Edmond, D. (2005). "Workflow Resource Patterns: Identification, Representation and Tool Support." In: *Advanced Information Systems Engineering*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, O. Pastor, and J. Falcão e Cunha. Vol. 3520. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 216–232. ISBN: 978-3-540-26095-0. DOI: 10.1007/11431855_16.

Sakurai, Y., Takada, K., Anisetti, M., Bellandi, V., Ceravolo, P., Damiani, E., and Tsuruta, S. (2012). "Toward sensor-based context aware systems." In: *Sensors* 12.1, pp. 632–649.

Sánchez González, L., García Rubio, F., Ruiz González, F., and Piattini Velthuis, M. (2010). "Measurement in business processes: A systematic review." In: *Business Process Management Journal* 16.1, pp. 114–134. ISSN: 1463-7154. DOI: 10.1108/14637151011017976.

Satyanarayanan, M. (2001). "Pervasive computing: Vision and challenges." In: *Personal Communications, IEEE* 8.4, pp. 10–17.

Scala, E., Micalizio, R., and Torasso, P. (2015). "Robust plan execution via reconfiguration and replanning." In: *AI Communications* 28.3, pp. 479–509. ISSN: 18758452. DOI: 10.3233/AIC-140629.

Scheer, A.-W. and Brabänder, E. (2010). "The Process of Business Process Management." In: *Handbook on Business Process Management 2*. Ed. by J. vom Brocke and M. Rosemann. Springer-Verlag Berlin Heidelberg, pp. 239–266.

Scholtz, B., Calitz, A., and Snyman, I. (2013). "The usability of collaborative tools." In: *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference on - SAICSIT '13*. Ed. by J. McNeill, K. Bradshaw, P. Machanick, and M. Tsietsi. New York, New York, USA: ACM Press, p. 347. ISBN: 9781450321129. DOI: 10.1145/2513456.2513503.

Schonenberg, H., Mans, R., Russell, N., Mulyar, N., and van der Aalst, W. M. P. (2008). "Process flexibility: A survey of contemporary approaches." In: *Advances in Enterprise Engineering I*. Springer, pp. 16–30.

Schönig, S., Cabanillas, C., Jablonski, S., and Mendling, J. (2015). "Mining the organisational perspective in agile business processes." In: *International Conference on Enterprise, Business-Process and Information Systems Modeling*, pp. 37–52.

Sedera, W., Rosemann, M., and Gable, G. G. (2002). "Measuring Process Modelling Success." In: *Proceedings of the 10th European Conference on Information Systems*, pp. 331–341.

Seeliger, A., Nolle, T., and Mühlhäuser, M. (Mar. 2017). "Detecting Concept Drift in Processes using Graph Metrics on Process Graphs." In: *Proceedings of the 9th International Conference on Subject-Oriented Business Process Management (S-BPM ONE '17)*.

Seethamraju, R. and Marjanovic, O. (2009). "Role of process knowledge in business process improvement methodology: A case study." In: *Business Process Mgmt Journal* 15.6, pp. 920–936. ISSN: 1463-7154. DOI: 10.1108/14637150911003784.

Serve, M., Yen, D. C., Wang, J.-.-C., and Lin, B. (2002). "B2B–enhanced supply chain process: toward building virtual enterprises." In: *Business Process Management Journal* 8.3, pp. 245–253. DOI: 10.1108/14637150210428952.

Shadish, W. R., Cook, T. D., and Campbell, D. T. (2002). *Experimental and quasi-experimental designs for generalized causal inference*. Belmont, CA: Wadsworth Cengage Learning. ISBN: 978-0395615560.

Shah, R. and Ward, P. T. (2007). "Defining and developing measures of lean production." In: *Journal of operations management* 25.4, pp. 785–805.

Shapiro, R., White, S. A., Bock, C., Palmer, N., zur Muehlen, M., Brambilla. Marco, and Gagné, D. (2012). *BPMN 2.0 handbook second edition: Methods, concepts, case studies and standards in business process management notation*. 2nd ed. Lighthouse Point, FLa.: Future Strategies. ISBN: 978-0-9849764-0-9.

Shoham, Y. and Tennenholtz, M. (1995). "On social laws for artificial agent societies: off-line design." In: *Artificial Intelligence* 73.1, pp. 231–252. ISSN: 0004-3702.

Siha, S. M. and Saad, G. H. (2008). "Business process improvement: Empirical assessment and extensions." In: *Business Process Management Journal* 14.6, pp. 778–802.

Simons, C. and Wirtz, G. (2007). "Modeling context in mobile distributed systems with the UML." In: *Visual Interactions in Software Artifacts* 18.4, pp. 420–439. ISSN: 1045-926X. DOI: 10.1016/j.jvlc.2007.07.001. URL: http://www.sciencedirect.com/science/article/pii/S1045926X07000377.

Skjoett–Larsen, T., Thernøe, C., and Andresen, C. (2003). "Supply chain collaboration." In: *International Journal of Physical Distribution & Logistics Management* 33.6, pp. 531–549. ISSN: 0960-0035. DOI: 10.1108/09600030310492788.

Škrinjar, R., Vukšić, V., and Štemberger, M. (2010). "Adoption of Business Process Orientation Practices: Slovenian and Croatian Survey." In: *Business Systems Research* 1.1-2, pp. 5–19. ISSN: 1847-9375. DOI: 10.2478/v10305-012-0022-0.

Smith, D. E., Frank, J., and Cushing, W. (2008). "The ANML language." In: *The ICAPS-08 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Smith, H. and Fingar, P. (2003). *Business process management: The third wave*. 1st ed. Tampa, Fla.: Meghan-Kiffer Press. ISBN: 0929652339.

Soffer, P. (2005). "On the notion of flexibility in business processes." In: *Proceedings of the CAiSE* 5, pp. 35–42.

Soffer, P., Kaner, M., and Wand, Y. (2012). "Towards Understanding the Process of Process Modeling: Theoretical and Empirical Considerations." In: *Business Process Management Workshops*. Ed. by W. M. P. van der Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, F. Daniel, K. Barkaoui, and S. Dustdar. Vol. 99. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 357–369. ISBN: 978-3-642-28107-5. DOI: 10.1007/978-3-642-28108-2_35.

Song, W., Jacobsen, H., Ye, C., and Ma, X. (Sept. 2016). "Process Discovery from Dependence-Complete Event Logs." In: *IEEE Transactions on Services Computing* 9.5, pp. 714–727. ISSN: 2372-0204. DOI: 10.1109/TSC.2015.2426181.

Stadtler, H., Kilger, C., and Meyr, H., eds. (2015). *Supply Chain Management and Advanced Planning*. Springer Texts in Business and Economics. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-55308-0. DOI: 10.1007/978-3-642-55309-7.

Štolba, M. and Komenda, A. (2013). "Fast-forward heuristic for multiagent planning." In: *Proceedings of the 1st Workshop on Distributed and Multi-Agent Planning*. Ed. by R. Nissim, D. L. Kovacs, and R. I. Brafman.

Strang, T. and Linnhoff-Popien, C. (2004). "A Context Modeling Survey." In: *Workshop Proceedings. First International Workshop on Advanced Context Modelling, Reasoning And Management at UbiComp*.

Swenson, K. D. (2010). "Mastering the Unpredictable." In: *How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done, Glossary*, pp. 314–317.

Sycara, K., Paolucci, M., Ankolekar, A., and Srinivasan, N. (2003). "Automated discovery, interaction and composition of semantic web services." In: *Web Semantics: Science, Services and Agents on the World Wide Web* 1.1, pp. 27–46. ISSN: 1570-8268.

Tax, N., Verenich, I., La Rosa, M., and Dumas, M. (2017). "Predictive Business Process Monitoring with LSTM Neural Networks." In: *Proceedings of the 29th International Conference on Advanced Information Systems Engineering (CAiSE)*. Ed. by E. Dubois and K. Pohl. Vol. 10253. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 477–492. ISBN: 978-3-319-59536-8. DOI: 10.1007/978-3-319-59536-8_30.

Taylor, F. (1911). *The Principles of Scientific Management*. Harper Brothers Publishers.

Tealeb, A., Awad, A., and Galal-Edeen, G. (2014). "Context-Based Variant Generation of Business Process Models." In: *Enterprise, Business-Process and Information Systems Modeling*. Springer, pp. 363–377.

Thakur, R. and Archana, T. (Apr. 2019). "Enabling Technologies and Applications of the Internet of Things." In: *Proceedings of Recent Advances in Interdisciplinary Trends in Engineering Applications (RAITEA)*. DOI: 10.2139/ssrn.3365534.

The Workflow Management Coalition Specification (1999). *Terminology & Glossary: WFMC-TC-1011 (Issue 3.0)*.

Thomas, O. and Fellmann, M. (2009). "Semantic Process Modeling – Design and Implementation of an Ontology-based Representation of Business Processes." In: *Business & Information Systems Engineering* 1.6, pp. 438–451. ISSN: 1867-0202. DOI: 10.1007/s12599-009-0078-8.

Torreño, A., Onaindia, E., and Sapena, Ó. (2012). "An approach to multi-agent planning with incomplete information." In: *Proceedings of 20th biennial European Conference on Artificial Intelligence (ECAI)* 242.762-767.

Torreño, A., Onaindia, E., and Sapena, Ó. (2014a). "FMAP: Distributed cooperative multi-agent planning." In: *Applied Intelligence* 41.2, pp. 606–626.

Torreño, A., Onaindia, E., and Sapena, Ó. (2014b). "Integrating individual preferences in multi-agent planning." In: *Proceedings of the 2nd Workshop on Distributed and Multi-Agent Planning*. Ed. by D. Borrajo, D. L. Kovacs, and A. Torreño.

Valentini, A., Micheli, A., and Cimatti, A. (2019). *Temporal Planning with Intermediate Conditions and Effects*. arXiv: 1909.11581 [cs.AI].

Van der Aalst, W. M. P. (1999). "Interorganizational Workflows: An Approach Based on Message Sequence Charts and Petri Nets." In: *Systems Analysis - Modelling - Simulation* 34.3, pp. 335–367.

Van der Aalst, W. M. P. (2004). "Business process management: A personal view." In: *Business Process Management Journal* 10.2, pp. 248–253. ISSN: 1463-7154. DOI: 10.1108/bpmj.2004.15710baa.001.

Van der Aalst, W. M. P. (2009). "Process-Aware Information Systems: Lessons to Be Learned from Process Mining." In: *Transactions on Petri Nets and Other Models of Concurrency II: Special Issue on Concurrency in Process-Aware Information Systems*. Ed. by K. Jensen and W. M. P. van der Aalst. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–26. ISBN: 978-3-642-00899-3. DOI: 10.1007/978-3-642-00899-3_1. URL: https://doi.org/10.1007/978-3-642-00899-3_1.

Van der Aalst, W. M. P. (2011). *Process mining: discovery, conformance and enhancement of business processes*. Springer Science & Business Media.

Van der Aalst, W. M. P. (2013). "Business process management: A comprehensive survey." In: *ISRN Software Engineering* 2013.

Van der Aalst, W. M. P. (2015). "Extracting Event Data from Databases to Unleash Process Mining." In: *BPM - Driving Innovation in a Digital World*. Ed. by J. vom Brocke and T. Schmiedel. Management for Professionals. Cham: Springer International Publishing, pp. 105–128. ISBN: 978-3-319-14429-0. DOI: 10.1007/978-3-319-14430-6_8.

Van der Aalst, W. M. P., Adriansyah, A., de Medeiros, A., Arcieri, F., Baier, T., Blickle, T., Bose, J., van den Brand, P., Brandtjen, R., and Buijs, J. (2012). "Process Mining Manifesto." In: *BPM 2011 Workshops, Part I, LNBIP 99*, pp. 169–194.

Van der Aalst, W. M. P., Barros, A. P., ter Hofstede, A. H. M., and Kiepuszewski, B. (2000a). "Advanced Workflow Patterns." In: *Cooperative Information Systems*. Ed. by P. Scheuermann and O. Etzion. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 18–29. ISBN: 978-3-540-45266-9.

Van der Aalst, W. M. P., Bichler, M., and Heinzl, A. (2018). "Robotic Process Automation." In: *Business & Information Systems Engineering* 60.4, pp. 269–272. DOI: 10.1007/s12599-018-0542-4.

Van der Aalst, W. M. P., Dreiling, A., Gottschalk, F., Rosemann, M., and Jansen-Vullers, M. H. (2006). "Configurable process models as a basis for reference modeling." In: *Business Process Management Workshops*. Ed. by C. J. Bussler and A. Haller. Springer, pp. 512–518. ISBN: 3540325956.

Van der Aalst, W. M. P. and Jablonski, S. (2000b). "Dealing with workflow change: identification of issues and solutions." In: *International Journal of Computer Systems Science & Engineering* 15.5, pp. 267–276.

Van der Aalst, W. M. P., Mylopoulos, J., Rosemann, M., Shaw, M. J., Szyperski, C., La Rosa, M., and Soffer, P., eds. (2013). *Business Process Management Workshops*. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-36284-2. DOI: 10.1007/978-3-642-36285-9.

Van der Aalst, W. M. P., Mylopoulos, J., Sadeh, N. M., Shaw, M. J., Szyperski, C., and Mendling, J. (2008). *Metrics for Process Models*. Vol. 6. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-540-89223-6. DOI: 10.1007/978-3-540-89224-3.

Van der Aalst, W. M. P., Pesic, M., and Schonenberg, H. (2009). "Declarative workflows: Balancing between flexibility and support." In: *Computer Science - Research and Development* 23.2, pp. 99–113.

Van der Aalst, W. M. P., Rubin, V., Verbeek, H. M., van Dongen, B. F., Kindler, E., and Günther, C. W. (2010). "Process mining: a two-step approach to balance between underfitting and overfitting." In: *Software & Systems Modeling* 9.1, pp. 87–111. ISSN: 1619-1366.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003). "Workflow Patterns." In: *Distributed and Parallel Databases* 14.1, pp. 5–51. ISSN: 0926-8782. DOI: 10.1023/A:1022883727209.

Van der Aalst, W. M. P. and van Hee, K. M. (2002). *Workflow Management: Models, methods and systems*. Cambridge, MA: MIT press.

Van der Aalst, W. M. P. and Verbeek, H. M. (2014). "Process discovery and conformance checking using passages." In: *Fundamenta Informaticae* 131.1, pp. 103–138.

Van der Aalst, W. M. P., Weijters, A. J. M. M., and Maruster, L. (2004). "Workflow mining: Discovering process models from event logs." In: *IEEE Transactions on Knowledge and Data Engineering* 16.9, pp. 1128–1142.

Van der Aalst, W. M. P., Weske, M., and Grünbauer, D. (2005). "Case handling: a new paradigm for business process support." In: *Data Knowledge Engineering* 53.2, pp. 129–162. ISSN: 0169-023X. DOI: 10.1016/j.datak.2004.07.003.

Van der Krogt, R., Bos, A., and Witteveen, C. (2002). "Replanning in a Resource-Based Framework." In: *Multi-Agent Systems and Applications II*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, V. Mařík, O. Štěpánková, H. Krautwurmová, and M. Luck. Vol. 2322. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 148–158. ISBN: 978-3-540-43377-4. DOI: 10.1007/3-540-45982-0_7.

Van der Krogt, R. and de Weerdt, M. (2005). "Plan Repair as an Extension of Planning." In: *ICAPS 2005: Proceedings of the 15th International Conference on Automated Planning and Scheduling, Monterey, California, USA, 5-10 June 2005*, pp. 161–170.

Van Beest, N., Kaldeli, E., Bulanov, P., Wortmann, J. C., and Lazovik, A. (2014). "Automated runtime repair of business processes." In: *Information Systems* 39, pp. 45–79. ISSN: 0306-4379. DOI: 10.1016/j.is.2013.07.003.

Van Dongen, B. F., Alves de Medeiros, A. K., and Wen, L. (2009). "Process Mining: Overview and Outlook of Petri Net Discovery Algorithms." In: *Transactions on Petri Nets and Other Models of Concurrency II*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan,

B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, K. Jensen, and W. M. P. van der Aalst. Vol. 5460. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 225–242. ISBN: 978-3-642-00898-6. DOI: 10.1007/978-3-642-00899-3_13.

Van Dongen, B. F., de Medeiros, A. K. A., Verbeek, H. M. W., Weijters, A. J. M. M., and van der Aalst, W. M. P. (2005). "The ProM Framework: A New Era in Process Mining Tool Support." In: *Applications and Theory of Petri Nets 2005*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, G. Ciardo, and P. Darondeau. Vol. 3536. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 444–454. ISBN: 978-3-540-26301-2. DOI: 10.1007/11494744_25.

Van Gorp, P. and Dijkman, R. M. (2013). "A visual token-based formalization of BPMN 2.0 based on in-place transformations." In: *Information and Software Technology* 55.2. Special Section: Component-Based Software Engineering (CBSE), 2011, pp. 365–394. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2012.08.014.

Vanhatalo, J., Völzer, H., and Koehler, J. (2008a). "The Refined Process Structure Tree." In: *Business Process Management*. Ed. by M. Dumas, M. Reichert, and M.-C. Shan. Vol. 5240. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 100–115. ISBN: 978-3-540-85757-0. DOI: 10.1007/978-3-540-85758-7_10.

Vanhatalo, J., Völzer, H., and Koehler, J. (2009). "The refined process structure tree." In: *Data & Knowledge Engineering* 68.9, pp. 793–818. DOI: 10.1016/j.datak.2009.02.015.

Vanhatalo, J., Völzer, H., Leymann, F., and Moser, S. (2008b). "Automatic Workflow Graph Refactoring and Completion: Service-Oriented Computing – ICSOC 2008." In: *Service-Oriented Computing – ICSOC 2008*. Ed. by A. Bouguettaya, I. Krueger, and T. Margaria. Vol. 5364. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 100–115. ISBN: 978-3-540-89647-0. DOI: 10.1007/978-3-540-89652-4_11.

Venable, J. R., Pries-Heje, J., and Baskerville, R. (2012). "A Comprehensive Framework for Evaluation in Design Science Research." In: *Design Science Research in Information Systems. Advances in Theory and Practice*. Ed. by K. Peffers, M. A. Rothenberger, and B. Kuechler. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 423–438. ISBN: 978-3-642-29862-2.

Verbeek, H. M. W., Pretorius, A. J., van der Aalst, W. M. P., and van Wijk, J. J. (2007). "Visualizing state spaces with Petri nets." In: *Computer Science Report* 7.01.

Verbeek, H. M. W. and van der Aalst, W. M. P. (2005). "Analyzing BPEL processes using Petri nets." In: *Proceedings of the Second International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management*, pp. 59–78.

Vom Brocke, J. (2009). "Design Principles for Reference Modelling." In: *Innovations in Information Systems Modeling*. Ed. by T. Halpin, J. Krogstie, and E. Proper. IGI Global, pp. 269–296. ISBN: 9781605662787. DOI: 10.4018/978-1-60566-278-7.ch014.

Vom Brocke, J. and Mendling, J., eds. (2018). *Business Process Management Cases: Digital Innovation and Business Transformation in Practice*. Management for Professionals. Cham: Springer International Publishing. ISBN: 978-3-319-58306-8. DOI: 10.1007/978-3-319-58307-5.

Vom Brocke, J., Schmiedel, T., Recker, J., Trkman, P., Mertens, W., and Viaene, S. (2014). "Ten principles of good business process management." In: *Business Process Management Journal* 20.4, pp. 530–548. ISSN: 1463-7154. DOI: 10.1108/BPMJ-06-2013-0074.

Wang, J., Zeng, C., He, C., Hong, L., Zhou, L., Wong, R. K., and Tian, J. (2012). "Context-aware role mining for mobile service recommendation." In: *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. Trento and Italy: ACM, pp. 173–178. ISBN: 978-1-4503-0857-1. DOI: 10.1145/2245276.2245310.

Wang, P., Ding, Z., Jiang, C., and Zhou, M. (2014). "Automated web service composition supporting conditional branch structures." In: *Enterprise Information Systems* 8.1, pp. 121–146. ISSN: 1751-7575. DOI: 10.1080/17517575.2011.584132.

Wang, W. and Brooks, R. J. (2007). "Empirical investigations of conceptual modeling and the modeling process." In: *Proceedings of the 2007 Winter Simulation Conference*. Ed. by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, pp. 762–770. DOI: 10.1109/WSC.2007.4419671.

Wang, X., Feng, Z., Huang, K., and Tan, W. (2017). "An automatic self-adaptation framework for service-based process based on exception handling." In: *Concurrency and Computation: Practice and Experience* 29.5. e3984 CPE-15-0402.R2, e3984. DOI: 10.1002/cpe.3984.

Wang, X. H., Zhang, D. Q., Gu, T., and Pung, H. K. (2004). "Ontology based context modeling and reasoning using OWL." In: *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pp. 18–22.

Weber, B., Reichert, M., and Rinderle-Ma, S. (2008a). "Change patterns and change support features–enhancing flexibility in process-aware information systems." In: *Data & Knowledge Engineering* 66.3, pp. 438–466.

Weber, I. (2007). "Requirements for Implementing Business Process Models through Composition of Semantic Web Services." In: *Enterprise Interoperability II*. Ed. by R. Gonçalves, J. Müller, K. Mertins, and M. Zelm. Springer London, pp. 3–14. ISBN: 978-1-84628-857-9. DOI: 10.1007/978-1-84628-858-6_1.

Weber, I., Hoffmann, J., and Mendling, J. (2008b). "Semantic business process validation." In: *Proceedings of the 3rd International Workshop on Semantic Business Process Management*.

Weijters, A., van der Aalst, W. M. P., and de Medeiros, A. A. (2006). "Process mining with the heuristics miner-algorithm." In: *Technische Universiteit Eindhoven, Tech. Rep. WP* 166.

Weske, M. (2012). *Business Process Management: Concepts, Languages, Architectures*. 2nd ed. Heidelberg Dordrecht London New York: Springer.

Wetzstein, B., Ma, Z., Filipowska, A., Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., and Cicurel, L. (2007). "Semantic Business Process Management: A Lifecycle Based Requirements Analysis." In: *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007) in conjunction with the 3rd European Semantic Web Conference (ESWC 2007)*.

Wohed, P., van der Aalst, W. M. P., Dumas, M., ter Hofstede, A. H. M., and Russell, N. (2006). "On the Suitability of BPMN for Business Process Modelling." In: *Business Process Management*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, S. Dustdar, J. L. Fiadeiro, and A. P. Sheth. Vol. 4102. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 161–176. ISBN: 978-3-540-38901-9. DOI: 10.1007/11841760_12.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Berlin and New York: Springer. ISBN: 978-3-642-29043-5.

Wooldridge, M. J. (2009). *An introduction to multiagent systems*. 2nd ed. Chichester: John Wiley & Sons Ltd.

Wu, K.-J., Tseng, M.-L., Chiu, A. S., and Lim, M. K. (2017). "Achieving competitive advantage through supply chain agility under uncertainty: A novel multi-criteria decision-making structure." In: *International Journal of Production Economics* 190, pp. 96–107.

Wynn, M. T., Verbeek, H. M. W., van der Aalst, W. M. P., ter Hofstede, A. H. M., and Edmond, D. (2009). "Business process verification-finally a reality!" In: *Business Process Management Journal* 15.1, pp. 74–92.

Ou-Yang, C. and Winarjo, H. (2011). "Petri-net integration – An approach to support multi-agent process mining." In: *Expert Systems with Applications* 38.4, pp. 4039–4051. ISSN: 09574174. DOI: 10.1016/j.eswa.2010.09.066.

Ye, J., Dobson, S., and McKeever, S. (2012). "Situation identification techniques in pervasive computing: A review." In: *Pervasive and mobile computing* 8.1, pp. 36–66.

Zairi, M. (1997). "Business process management: a boundaryless approach to modern competitiveness." In: *Business Process Management Journal* 3.1, pp. 64–80.

Zamperoni, A. and Löhr-Richter, P. (1993). "Enhancing the Quality of Conceptual Database Specifications through Validation." In: *Proceedings of the 12th International Conference on Entity-Relationship Approach (ER'93)*, pp. 87–99.

Zhang, D., Adipat, B., and Mowafi, Y. (2009). "User-Centered Context-Aware Mobile ApplicationsâThe Next Generation of Personal Mobile Computing." In: *Communications of the Association for Information Systems* 24.1, p. 3. ISSN: 1529-3181.

Zhang, J. (2017). "The Technical Foundation of a Multi-Agent System." In: *Multi-Agent-Based Production Planning and Control*. John Wiley & Sons Singapore Pte. Ltd, pp. 21–53. ISBN: 9781118890073. DOI: 10.1002/9781118890073.ch2.

Zhang, Z. and Sharifi, H. (2000). "A methodology for achieving agility in manufacturing organisations." In: *International Journal of Operations & Production Management* 20.4, pp. 496–513.

Zheng, C., Wen, L., and Wang, J. (2017). "Detecting Process Concept Drifts from Event Logs." In: *On the Move to Meaningful Internet Systems. OTM 2017 Conferences*. Ed. by H. Panetto, C. Debruyne, W. Gaaloul, M. Papazoglou, A. Paschke, C. A. Ardagna, and R. Meersman. Cham: Springer International Publishing, pp. 524–542. ISBN: 978-3-319-69462-7.

Zheng, X. and Yan, Y. (2008). "An Efficient Syntactic Web Service Composition Algorithm Based on the Planning Graph Model." In: *2008 IEEE International Conference on Web Services (ICWS)*, pp. 691–699. DOI: 10.1109/ICWS.2008.134.

Zhou, J., Gilman, E., Palola, J., Riekki, J., Ylianttila, M., and Sun, J. (2011). "Context-aware pervasive service composition and its implementation." In: *Personal and ubiquitous computing* 15.3, pp. 291–303. ISSN: 1617-4909. DOI: 10.1007/s00779-010-0333-5.

Zhu, X., Recker, J., Zhu, G., Santoro, F. M., and Al-Mashari, M. (2014a). "Exploring location-dependency in process modeling." In: *Business Process Management Journal* 20.6.

Zhu, X., Zhu, G., Vanthienen, J., Baesens, B., et al. (2014b). "Towards location-aware process modeling and execution." In: *Business Process Management Workshops*.

Zur Muehlen, M. and Ho, D. T.-Y. (2006). "Risk Management in the BPM Lifecycle." In: *Business Process Management Workshops*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. J. Bussler, and A. Haller. Vol. 3812. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 454–466. ISBN: 978-3-540-32595-6. DOI: 10.1007/11678564_42.

Zur Muehlen, M. and Rosemann, M. (2004). "Multi-Paradigm Process Management." In: *Proceedings of CAiSE'04 Workshops - 5th Workshop on Business Process Modeling, Development and Support (BPMDS 2004)*. Ed. by J. Grundspenkis and M. Kirikova, pp. 169–175.

7
Appendices

7.1 Paper 1: Automated Planning of Process Models: The Construction of Simple Merges

7.1.1 Pseudocode of the Algorithm

Listing 7.1 Pseudocode of the procedure ALGORITHM

```
1  procedure ALGORITHM(A)
2    AllChoiceNodes={}
3    AllParallelNodes={}
4    MARKACTIONS(A)
5  end
```

Listing 7.2 Pseudocode of the procedure MarkActions(A)

```
1  procedure MarkActions(A)
2    ChoicePrecedings = {}
3    SubLength = A.length
4    while a := A.pop()
5      Precedings = {}
6      switch (a.type){
7        case JoinNode:
8          precedingTokens = PrecedingNode(a).tokens
9          a = MarkParallels(a)
10         a.tokens.add(precedingTokens)
11         Precedings.add(PrecedingNode(a))
12         AddUniqueToken(a)
13         AllParallelNodes.add(a)
14         break
15       case ChoiceNode:
16         if (PathsNotReadyYet())
17         break
18         endif
19         MarkChoice(a)
20         AddUniqueToken(a)
21         forall b ∈ a.branches
22         a.tokens.add(b[0].tokens)
23         endfor
24         AllChoiceNodes.add(a)
25         MergePaths(a)
26         ChoicePrecedings.add(PrecedingNode(a))
27         break
28       case Action:
29         Precedings.add(PrecedingNode(a))
30         EqualActions = {}
31         EqualActions.add(a)
32         forall b ∈ A
```

```
33              if (a == b && getUniqueToken(SucceedingNode(a)) ==
                    getUniqueToken(SucceedingNode(b)))
                    Precedings.add(PrecedingNode(b))
34                EqualActions.add(b)
35                A.remove(b)
36              endif
37            endfor
38            uniqueToken=null
39            if (SubLength != EqualActions.length)
40              uID = GenerateUniqueID()
41              uniqueToken = new Token(uID)
42            endif
43            forall b ∈ EqualActions
44              forall tok ∈ SucceedingNode(b).tokens
45                b.tokens.add(tok)
46              endfor
47              b.tokens.add(uniqueToken)
48            endfor
49            break
50        if (Precedings.length > 0)
51          MarkActions(Precedings)
52        endif
53      endwhile
54      if (ChoicePrecedings.length > 0)
55        MarkActions(ChoicePrecedings)
56      endif
57    end
```

Listing 7.3 Pseudocode of the procedure MarkChoice(a)

```
1   procedure MarkChoice(a)
2     choiceConstruct = ChoiceConstruct.new
3     forall b ∈ a.branches
4       choiceConstruct.add({b.conditions, b.actions[0].tokens})
5     endfor
6     a.choiceConstruct = choiceConstruct
7     a.uniqueToken = null
8   end
```

Listing 7.4 Pseudocode of the procedure MarkParallels(node)

```
1  procedure MarkParallels(node)
2    parallelConstruct = ParallelConstruct.New
3    forall a ∈ PrecedingNodes(node)
4      Branch = {}
5      while a != ParallelNode
6        if (a == JoinNode)
7          a = MarkParallels(a)
8        endif
9        Branch.add(a)
10       a = PrecedingNode(a)
11     endwhile
12     parallelConstruct.add(Branch)
13   endfor
14   a.parallelConstruct = parallelConstruct
15   a.uniqueToken = null
16   return a
17 end
```

Listing 7.5 Pseudocode of the procedure getUniqueToken(node)

```
1  procedure getUniqueToken(node)
2    switch node
3      case ChoiceNode
4        return node.uniqueToken
5      case ParallelNode
6        return node.uniqueToken
7      case Action
8        return node.tokens.last
9  end
```

Listing 7.6 Pseudocode of the procedure addUniqueToken(node)

```
1  procedure addUniqueToken(node)
2    boolean existsEqual = false
3    switch node
4      case ChoiceNode
5        forall choNode ∈ AllChoiceNodes
6          if (node.choiceConstruct == choNode.choiceConstruct)
7            node.uniqueToken = getUniqueToken(choNode)
8            existsEqual = true
9          endif
10       endfor
11       break
12     case ParallelNode
13       forall paraNode ∈ AllParallelNodes
14         if (node.parallelConstruct == paraNode.parallelConstruct)
15           node.uniqueToken = getUniqueToken(ParaNode)
16           existsEqual = true
17           endif
18       endfor
19       if !existsEqual
20         uID=generateUniqueID()
21         node.uniqueToken = token(uID)
22         node.tokens.add(token(uID))
23       endif
24       return
25 end
```

Listing 7.7 Pseudocode of the procedure mergePaths(choiceNode)

```
1  procedure mergePaths (choiceNode)
2    NodeArray = {}
3    forall b ∈ choiceNode.branches
4      NodeArray.add(b.actions[0])
5    endfor
6    mergeBranches (NodeArray)
7  end
```

Listing 7.8 Pseudocode of the procedure mergeBranches(NodeArray)

```
1  procedure mergeBranches (NodeArray)
2    TokenArray = new Array[NodeArray.length]
3    for (i=0; i < NodeArray.length; i++)
4      b = NodeArray[i]
5      tok = smallestToken(b) // at least T(0)
6      TokenArray[i] = tok
7      for (j=0; j < NodeArray.length; j++)
8        if (j != i)
9          c = NodeArray[j]
10         comp = max(b.SameTokens(c)) // biggest element in both b.Tokens and
               c.Tokens
11         if (tok < comp)
12           tok = comp
13         endif
14       endif
15     endfor
16     if (TokenArray[i] < tok)
17     TokenArray[i] = tok
18     endif
19   endfor
20   tok = max(TokenArray) // highest entry in Array.. will appear more than
       once
21   MergeArray = {}
22   IndexArray = {}
23   for (i = 0; i < TokenArray.length; i++)
24     if (TokenArray[i] == tok)
25       MergeArray.add(NodeArray[i])
26       IndexArray.add(i)
27     endif
28   endfor
29   NewMerge = CreateMergeNode(MergeArray, tok)
30   NodeArray.add(SucceedingNode(NewMerge))
31   NodeArray.deleteIndices(IndexArray)
32   if (NodeArray.length > 1)
33     mergeBranches (NodeArray)
34   endif
35 end
```

Listing 7.9 Pseudocode of the procedure CreateMergeNode(MergeArray,tok)

```
1  procedure CreateMergeNode(MergeArray,tok)
2    mergeNode = new MergeNode
3    forall node  MergeArray
4      while (getUniqueToken(node)) != tok)
5        if !(SucceedingNode(node).isChoiceNode)
6          node = SucceedingNode(node) //If the algorithm reaches a
               ParallelNode, the SucceedingNode-function returns the succeeding
               Node of the JoinNode
7          if (getUniqueToken(node) == tok && node.isMergeNode)
8            mergeNode = node;
9            node = SucceedingNode(node)
10         endif
11       else
12         forall b ∈ node.branches
13           if (tok ∈ getAllTokens(b[0]))
14           node = b[0]
15           endif
16         endfor
17       endif
18     endwhile
19   endfor
20   forall b ∈ MergeArray
21     mergeNode.precedings.add(PrecedingNode(b)) //If preceding node of b is a
         mergeNode, the PrecedingNode-function returns the preceding nodes of
         the mergeNode
22   endfor
23   mergeNode.succeeding = MergeArray[0]
24   iteratingNode = MergeArray.pop()
25   while bs = getBelieveStateAfter(iteratingNode)
26     forall a ∈ MergeArray
27       bs = bs ∪ getBelieveStateAfter(a)
28     endfor
29     iteratingNode = SucceedingNode(iteratingNode)
30     for (i = 0; i < MergeArray.length; i++)
31       MergeArray[i] = SucceedingNode(MergeArray[i])
32     endfor
33   endwhile
34   return mergeNode
35 end
```

Listing 7.10 Pseudocode of the procedure PathsNotReadyYet(Node)

```
1 procedure PathsNotReadyYet(Node)
2   forall b ∈ Node.branches
3     if b[0].Tokens == {}
4       return true
5     endif
6   endfor
7   return false
8 end
```

7.1.2 Mathematical Evaluation of the Algorithm

7.1.2.1 Termination

The ALGORITHM procedure terminates:

The lines 2 and 3 terminate obviously, so it suffices to show that the MarkActions procedure starting in line 4 terminates.

The MARKACTIONS procedure terminates:

This is shown by proving that the number of iterations of each loop is finite, and that each statement of the algorithm (also the recursions in line 52 and line 56) terminates.

The while-loop starting in line 4 terminates:

- The statements in the while-loop are only executed for a finite number of elements, because A is finite.
- The statement in line 5 terminates obviously as it is a simple (set) operation.
- The switch-case statement starting in line 6 also terminates: The cases *JoinNode* and *ChoiceNode* terminate due to the fact that the procedures MarkParallels, AddUniqueToken, PathsNotReadyYet, MarkChoice, MergePaths terminate (see Lemmata) and the rest of the statements are trivially terminating set operations. The case Action terminates as well: The statements in the lines 29-31 are again simple set operations. The for-loop starting in line 32 terminates because A is finite and the statements in lines 34-36 terminate trivially. The statements in the lines 39-42 terminate obviously. The for-loops starting in line 44 and 45 terminate since the sets *EqualActions* and *SucceedingNode*().*Tokens* are finite and the statements in line 46 and 48 are simple set operations.

- The recursion in line 52 terminates because it is only invoked when the length of the set *Precedings* is non-zero. This happens only a finite number of times: The set *Precedings* starts empty (line 5). An element can only be added if there is a preceding node (cf. lines 11, 29, 34). This only occurs a finite number of times because every time the recursion is invoked, the algorithm goes one layer upward on the path. However, our paths are assumed to be finite.
- Furthermore, the set *Precedings* is always finite because there are only a finite number of preceding nodes for every node and only the preceding nodes of a finite number of nodes are considered.
- Finally, the recursion in line 56 terminates: It is only called upon when the length of the set *ChoicePrecedings* is non-zero. To prove that this occurs only a finite number of times, our argumentation is the following: The set *ChoicePrecedings* starts empty (line 2). An element can only be added if there is a preceding node (cf. line 26), and this only occurs a finite number of times because every time the recursion is invoked, the algorithm goes one layer upward on the path. However, our paths are assumed to be finite.
- Additionally, the set *ChoicePrecedings* is always finite because there is only one preceding node for every node and only the preceding nodes of a single node are considered.

Lemma 1: MARKPARALLELS terminates:

To show that MarkParallels terminates, it suffices to prove that the for-loop starting in line 3 terminates since the rest of the statements are obviously terminating. The for-loop is only called upon a finite number of times because there is only a finite number of preceding nodes for each node. Each iteration of the for-loop is finite, because the statements in the lines 4 and 12 are trivially terminating and the while-loop in line 6 gets called upon only a finite number of times. The reasoning for this is as follows: We may assume that *a* is not a *ParallelNode* (otherwise the while-loop doesn't get invoked at all). If *a* is not a JoinNode, we move up one layer on our path because of line 10. This "moving-up" can only occur a finite number of times due to the assumption that our paths are finite. So we only have to consider the fact that *a* is a JoinNode. Then MarkParallels gets invoked again, but with a node that is one layer upward compared to the initial invocation. Because our paths are assumed to be finite, also this moving upward must come to an end. Note that the lines 9 and 10 terminate obviously.

Lemma 2: AddUniqueToken terminates:

Only the for-loops starting in the lines 5 and 13 need to be considered, the rest of the operations are simple set operations. These for-loops contain only simple set operations as well and are called upon only a finite number of times because the sets *AllChoiceNodes* and *AllParallelNodes* are finite: The sets start out empty (cf. lines 2 and 3 in Algorithm). An element can only be added to them via the MarkActions primitive, in lines 24 resp. 13, and just one at most per iteration of MarkActions. However, we have already seen that the MarkActions procedure gets invoked only a finite number of times.

Lemma 2a: GetUniqueToken terminates:

Obvious.

Lemma 3: PathsNotReadyYet terminates:

This is clear since the for-loop in line 2 terminates, because there are only finitely many branches descending from each ChoiceNode.

Lemma 4: MarkChoice terminates:

To prove that MarkChoice terminates, it is sufficient to prove that the for-loop starting in line 3 terminates because the rest of the statements are trivially terminating. The for-loop is only invoked a finite number of times since there are only a finite number of branches after a choice node.

Lemma 5: MergePaths terminates:

MergePaths terminates, if MergeBranches in line 6 terminates, because the for-loop in line 3 is finite since every choice node only has a finite number of branches.

Lemma 6: MergeBranches terminates:

- The given *NodeArray* in line 1 is finite (as shown later). That causes the for-loops in lines 3 and 7 to be invoked only a finite number of times. All the other statements inside the for-loop starting in line 3 and also in the lines 20-22 are simple set operations. The next for-loop in line 23 terminates since *TokenArray* has the same length as *NodeArray* and the for-loop only contains simple set operations. Lines 29-31 terminate as CreateMergeNode terminates (see Lemma) and the other lines are simple set operations.
- The MergeBranches function terminates, if the recursive call in line 33 happens only a finite number of times, what will be shown in the following:

- The *NodeArray* is finite in line 33: It was finite in the initial call of the MERGE-BRANCHES function (line 6 in MergePaths), owing to the fact that a choice node only has a finite number of branches. It hasn't been changed before line 30. Furthermore, even though one node is being added to the *NodeArray* in line 30 and thus the length increases by one, in the next step in line 31 more than one node are being removed. This is caused by the fact that the length of *IndexArray* is greater than one. Hence, the length of *NodeArray* decreases in the lines 30 and 31 in total and therefore it decreases in every recursive call of the function, until its length is 1 and the recursion stops.

Lemma 7: CreateMergeNode terminates:

The for-loop starting in line 3 iterates only finitely many times, because *MergeArray* is finite as a subset of the finite set *NodeArray* (cf. lines 21, 25 in MergeBranches).

- The while-loop starting in line 4 iterates finitely many times as well: When the unique token of the current node is not *tok*, one of the following cases can occur:
  - The succeeding node is not a choice node: In this case the algorithm traverses further to the next succeeding node.
  - The succeeding node is a choice node: Then one of the first nodes of the branches of the choice node is marked with the token *tok*, and the algorithm also proceeds to this succeeding node that is marked with *tok* (see line 14). Because our paths are finite, this moving on can only occur a finite number of times, until the algorithm reaches a node whose highest token (i.e. uniqueToken) is *tok*.
- The operations within the while-loop starting in line 4 terminate, as they are simple set operations except for the for-loop starting in line 12, which terminates because a choice node only has finitely many branches.

These statements patched together allow us to conclude that the for-loop starting in line 3 terminates.

The for-loop starting in line 20 terminates, because *MergeArray* is finite, as we have already seen, and line 21 is a trivially terminating set operation.

- The lines 23 and 24 terminate obviously.
- The for-loop starting in line 26 terminates, because line 27 is a simple set operation and *MergeArray* is finite (it was finite before and no element has been added).

- Line 29 is trivially terminating.
- The for-loop starting in line 30 terminates, because it only contains a simple set operation and we know *MergeArray* is finite.
- The while-loop in line 25 gets invoked only a finite number of times, because in each iteration one proceeds to a succeeding node in line 29 and our paths are assumed to be finite.

These statements allow us to conclude that the while-loop from line 25 terminates. This finishes the proof of the termination of CreateMergeNode.

7.1.2.2 Completeness and Correctness (proof sketch)

The algorithm identifies all mergeable paths and does not create wrong simple merges:
We prove this theorem by proving the following two statements:

1) A node *a* can only get the same unique token as a node *b*, if node *a* is in the same equality groups as node *b* (short and in quantifiers: $(UniqueToken\,(a) = UniqueToken\,(b)) \Rightarrow (\forall\, eg \in EG : a \in eg \iff b \in eg)$).This leads to no incorrect merges being done.

2) If two nodes *a* and *b* are in the same equality groups, then *a* and *b* get the same token
(short and in quantifiers: $(\forall\, eg \in EG : a \in eg \iff b \in eg) \Rightarrow (UniqueToken\,(a) = UniqueToken\,(b))$). This leads to all correct merges being done.

Concerning 1):
Let us assume there exist two nodes *a* and *b* which have the same unique token.
Case a): The nodes *a* and *b* are choice nodes (parallel nodes)

> A choice node (parallel node) gets its token via AddUniqueToken. As seen in lines 5-10 (lines 13-18) of AddUniqueToken, such a node gets the same token as an existing node, if and only if they have the same choice constructs (parallel constructs), otherwise they get a new unique token (line 19-23). Having the same choice constructs (parallel constructs) means that they have the same following branches with the same conditions, as can be seen in the MarkChoice (MarkParallels) procedure. Thus *a* and *b* are in the same equality groups.

Case b): The nodes *a* and *b* are both actions

An action gets its unique token in line 48 in the MarkActions primitive. Because of the structure of the algorithm, the nodes *a* and *b* having the same unique token means that either

b1) "*a* and *b* on the same layer"

Both *a* and *b* are in the same *EqualActions*-set. This means that the actions have the same effect and their succeeding nodes are equal. So they are in the same equality groups.

b2) "*a* and *b* not on the same layer"

The actions *a* and *b* are not in the same *EqualActions*-set, but they have the same unique token nevertheless. Without loss of generality we may assume that *a* got its tokens first (let us say, in iteration i_a) and the tokens of *b* were assigned later (in iteration i_b). Then in all the iterations between i_a and i_b, *SubLength* was equal to *EqualActions.length* (otherwise a new unique token would have been generated, cf. lines 40-43 in MarkActions). This means that *a* and *b* are in the same equality groups.

Case c): The nodes *a* and *b* are two different kinds of nodes

If a node is a choice node (parallel node), it only gets the same unique token as existing choice nodes (parallel nodes), or it gets a new unique token, but never the same unique token as an already existing action node, or a parallel node (choice node) (see proof of case a) for details). On the other hand, later the same tokens that a choice node (parallel node) already possesses may be assigned to an action node, but in this case, the argumentation of case b2) shows that then the action node and the choice node (parallel node) are in the same equality groups.

Concerning 2):

We may assume that the nodes *a* and *b* are in the same equality groups.

Case a): The nodes *a* and *b* are choice nodes (parallel nodes)

When *a* and *b* are in the same equality groups, their following branches and those conditions are the same. This results in their choice constructs (parallel constructs), which are constructed in MarkChoice (MarkParallels) being equal. Thus the lines 5-10 (lines 13-18) in AddUniqueToken result in *a* and *b* getting the same token.

Case b): The nodes *a* and *b* are both actions

They are in the same equality groups, which can mean either

b1) "*a* and *b* on the same layer"

The actions *a* and *b* have the same effect and their succeeding nodes are equal. Then both, *a* and *b*, are added to the array *Precedings* in the same iteration (cf. lines 29, 34 in MarkActions). So there is an invocation of MarkActions(A) where A contains both *a* and *b*. When either *a* or *b* gets added to the array *EqualActions*, the other one gets as well (see lines 31-33, 35 in MarkActions). Lines 44 and onward show that this results in marking *a* and *b* with the same token.

b2) "*a* and *b* not on the same layer"

Without loss of generality we may assume that *a* gets its tokens first (let us say, in iteration i_a) and the tokens of *b* are assigned later (in iteration i_b). Because *a* is in all equality groups that *b* is in, in all iterations of MarkActions between i_a and i_b, *SubLength* and the length of the (in this case unique) *EqualActions*-set cannot differ, because otherwise a new equality group would emerge, containing *b* but not *a*. This results in node *b* (and all nodes between *a* and *b*) acquiring all equality tokens from node *a* via the lines 44-49 in MarkActions, but not gaining a different unique token, as can be seen in the lines 39-43 in MarkActions.

Case c): The nodes *a* and *b* are different kinds of nodes

The nodes *a* and *b* are different kinds of nodes: A parallel node and a choice node cannot have all their equality groups in common, so we only have to account for the case of an action node and a choice node (parallel node) being in the same equality groups. This case can only occur when the choice node (parallel node) is a (not necessarily directly) succeeding node of the action node. The argumentation is very similar to the one in case b2). Without loss of generality, let *a* be the action node and *b* be the choice node (parallel node), and let i_a and i_b be the iterations of MarkActions in which their tokens are assigned, respectively. Because *b* is in all equality groups that *a* is in, in all iterations of MarkActions between i_b and i_a, *SubLength* and the length of the (in this case unique) *EqualActions*-set cannot differ, because otherwise a new equality group would emerge, containing *a* but not *b*. This results in node *a* (and all nodes between *b* and *a*) acquiring all equality tokens from node *b* via the lines 44-49 in MarkActions, but not gaining a different unique token, as can be seen in the lines 39-43 in MarkActions.

7.1.2.3 Minimality (proof sketch)

The algorithm creates a simple merge as early as possible:

A set of flow nodes only has the same tokens, caused by the algorithm, if they belong to the same equality group. As shown in the proof of correctness, the algorithm finds all flow nodes that belong to the same equality group and marks them with tokens accordingly. This means, the result is minimal, if the algorithm finds not only a *correct* possible point to merge branches with nodes with the same unique tokens, but the *earliest one*.

MergeBranches is invoked with a set of flow nodes following a specific exclusive choice (cf. lines 3-5 of MergePaths). It identifies the highest token *tok*, with which at least two outgoing branches of this exclusive choice are marked (cf. lines 3-20). As the tokens grow with the upwards traversal, this is the "largest" set of flow nodes (considering the length of the sequence of flow nodes) in the same equality group succeeding an exclusive choice. We thereby identify the earliest possible point in the process models, where at least two outgoing branches could be merged.

createMergeNode now traverses down each branch following the nodes in *MergeArray*, until it reaches the flow nodes with the previously identified maximal token *tok* as their highest token (lines 4-18) and then creates a simple merge before these flow nodes. To be precise, it inserts a *MergeNode* whose predecessors are the former predecessors (cf. line 21) of the flow nodes in *MergeArray* and whose successor is the first flow node within the identified equality group (cf. line 23). This leads to the fact that the first (earliest) possible merge position is found and thus the result is minimal.

As we recursively invoke MergeBranches again (cf. lines 32-34) if there are unmerged outgoing branches, we ensure to repeat this identification for each outgoing branch of the exclusive choice and thus we create a simple merge "as early as possible" for each outgoing branch. This leads to a minimal process model.

7.1.2.4 Computational Complexity (sketch)

The main method of the algorithm is MarkActions, which is invoked recursively. It identifies equality groups in one layer of the process model by traversing (line 4) the flow nodes in one layer and invokes itself again with the predecessors of the flow nodes within these equality groups (line 52 and 56). As every flow node could be preceded by only one other flow node, the method invokes itself n times in the maximum, whereas n is denoted as the maximum of the number of goal belief states as this is equal with the number of flow nodes in the last layer and the number of sequential flow nodes in the longest path of the planning graph.

In an average case, the algorithm should be much more efficient, as the number of flow nodes in one layer decreases when reaching exclusive choices.

In the case, that all actions preceding the goal belief states are equal, the MarkActions primitive gets invoked one time for each layer as it identifies one overall equality group. Assuming, that the subsequent flow nodes stay equal until the initial belief state, the number of flow nodes is constant for each layer. Thus, the method invokes itself n times, whereas n is the number of layers in the planning graph. At the other end, if all n actions preceding the goal belief states differ, the MarkActions primitive is invoked n times for the next layer. As then each subset only contains one action, the nested while loop is iterated only once for each invocation. Thus, the code within the while loop is executed n times. Let us assume that in each layer the algorithm finds two equality groups with equal size. Thus, MarkActions gets invoked 2 times in each layer, while the length of the *Preceding* subset decreases to n/2 with each iteration. Thus, the maximum number of executions of the code within the while loop is constantly n in all cases.

Within the while loop we need to consider a switch, which implies to use the case with maximum complexity for the further calculations. The case "ChoiceNode" is the most complex, as the outgoing branches of an exclusive choice get merged when reaching it. The most complex part of this case is the invocation of MergePaths, which further invokes MergeBranches for the outgoing branches. MergeBranches iterates in two nested for loops over the Array of outgoing branches which results in a computational complexity of $O(n^2)$ with n as the number of outgoing branches of the exclusive choice. Further, it invokes CreateMergeNode. This primitive traverses each outgoing branch of the exclusive choice that could be merged, too. Further, it traverses along the branches until the first flow nodes of the identified equality group. Additionally, we need to traverse the outgoing branches of a nested exclusive choice, if present. Thus, the overall computational complexity of this primitive is $O(n^3)$.

As the code within the while loop of MarkActions is executed n times at most and the most complex case "ChoiceNode" has a computational complexity of $O(n^3)$, the complete algorithm has an overall complexity of $O(n^5)$.

7.1.3 Verification Properties of constructed Planning Graphs

7.1.3.1 Soundness of the resulting Planning Graphs

Following van der Aalst (1998), it needs to be proven, that (1) for each belief state of the planning graph that could be reached from the initial state ($bs \in Part_{\text{BS}}$) a sequence of actions or control flow structures exist that leads from the belief state to a goal state, that (2) the instance of the process terminates, if an edge, resulting in a goal state is traversed, and that (3) there are no "dead actions", which means, for each action at least one feasible path exists that contains this action.

When considering an initial planning graph, constructed by means of an automated

planning approach, this initial planning graph fulfils (1) and (3) obviously, as automated planning approaches like Heinrich et al. (2011) construct correct (cf. (1)) and minimal (i.e., there are no actions that could not be executed, cf. (3)) planning graphs. As our approach does not construct any new action, (1) and (3) is furthermore guaranteed for the planning graph resulting after construction of simple merges in an automated manner.

As our approach constructs simple merges so that the goal belief state has exactly one incoming edge and no outgoing edges, (2) is also guaranteed for the planning graph that is constructed by means of our approach.

In sum: Our approach constructs sound planning graphs if for each belief state of the initial planning graph that could be reached from the initial state ($bs \in Part_{\text{BS}}$) a sequence of actions or control flow structures exist that leads from the belief state to a goal state (1) and the initial planning graph does not contain dead actions (3).

7.1.3.2 S-Coverability of the resulting Planning Graphs

In order to ensure that the resulting planning graphs are covered by S-components, we need to ensure that the constructed planning graphs are (1) well-formed and (2) free choice.

Ad (1): Assuming a well-formed planning graph as the starting point for our approach then our approach constructs a well-formed planning graph too. This is true, as the constructed simple merges do not increase the number of reachable states and as they do not influence liveness (i.e., no transitions, actions in our terms, are added and states are only unified but no new states are added) of the planning graph.

Ad (2): Considering Definition 1, the transition function $R: BS \times A \rightarrow 2^{\text{BS}}$ associates to each belief state $bs \in BS$ and to each action $a \in A$ the set $R(bs, a) \subseteq BS$ of next belief states. Thus, each action (related to transitions in Petri nets) is connected to exactly one preceding belief state and has exactly one input edge (arc). According to Verbeek *et al.* (2001), process models (they refer to workflows) are free choice if for every two actions, the preconditions are either disjoint or identical. In our context of automated process planning, this issue is addressed by means of a control flow structure called R-XOR. The control flow structure R-XOR is used to denote two or more feasible solutions, which are functional equivalents to represent a particular (sub)structure and behavior in a process model (for details, cf. Heinrich *et al.*, 2015c, pp. 9-10). However, the control flow structure R-XOR is inferred at runtime, which means, the conditions to select exactly one outgoing path out of a set of outgoing actions are only given at runtime. This selection could be done, for instance, on non-functional properties that are unknown or undefined at planning time.

7.1.4 Additional Considerations

Within our paper, we considered the creation of pattern-compounds as they represent well-formed and sound block-structured fragments of a process model. Thus, process models consisting of such pattern-compounds are more readable and therefore understandable for laymen.

In some cases, it could be favorable to reduce the amount of duplicate actions while ignoring pattern-compounds as then, even more duplicates could be removed. To enable this, only minor adaptions have to be performed. Considering this requirement, it is no longer sufficient to only use the token of nested exclusive choices to identify mergeability. Instead, choice constructs of nested exclusive choices have to be considered within MergeBranches:

- Within the comparison (lines 3-19) in MergeBranches choice constructs of nested exclusive choices need to be compared in detail instead of just using the tokens of exclusive choices as a criterion for mergeability.
- The termination criterion of the while loop in line 4 of CreateMergeNode has to be adapted to consider equal actions in outgoing branches of nested exclusive choices

7.2 Paper 2: Automated Planning of context-aware Process Models

7.2.1 Full Version of Figure 2.7

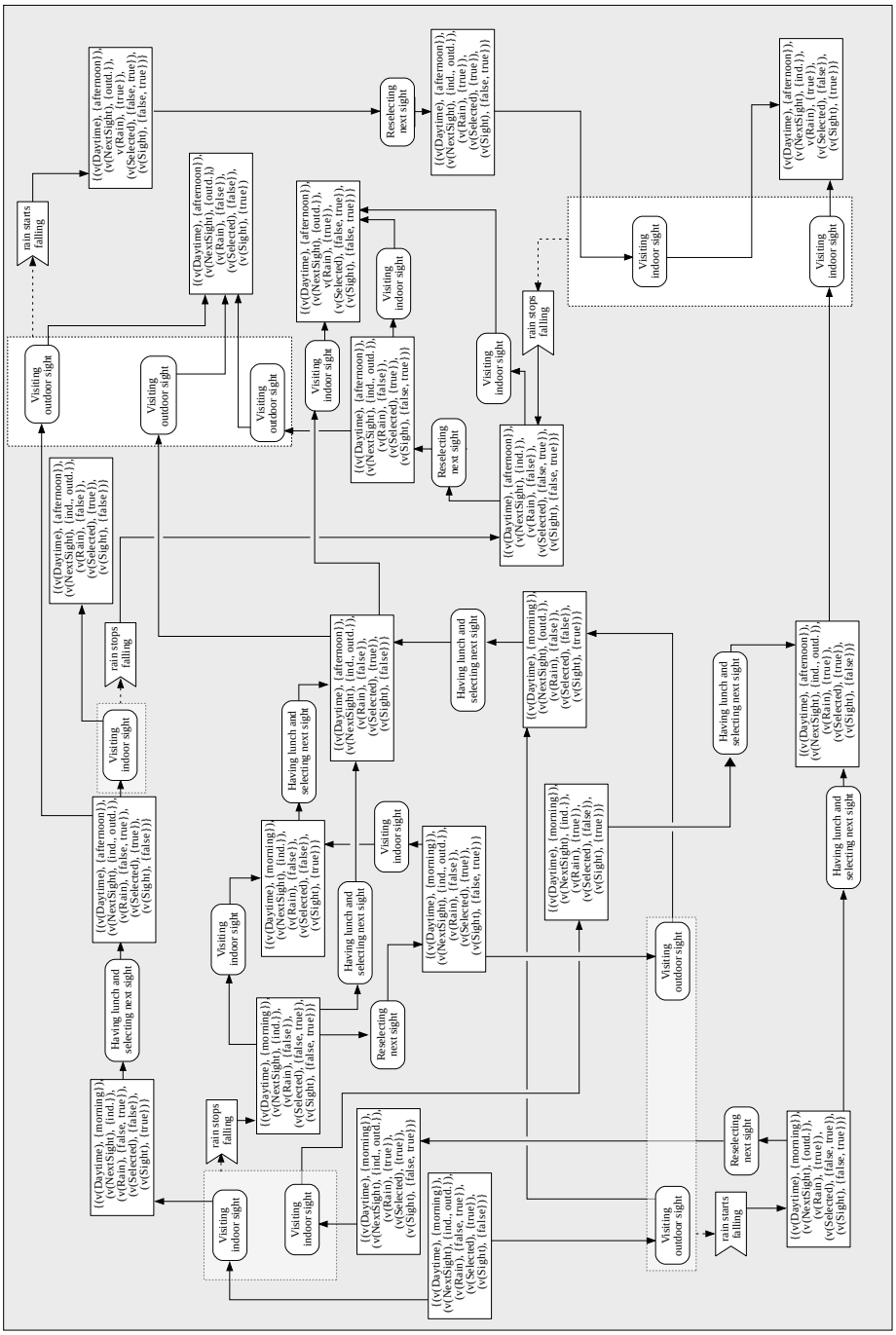

Figure 7.1: Full Version of Figure 2.7

7.2.2 Pseudocode of the Algorithm

Listing 7.11 Pseudocode of the class Algorithm

```
1  class Algorithm {
2    generateGraph(){
3      recursiveCreateGraph(initialState);
4    }
5    boolean recursiveCreateGraph(State state){
6      if (state.isGoalState() || state.isAlreadyChecked()){
7        return true
8      } else {
9        List Action applicableActions = state.getApplicableActions();
10       if (applicableActions.isEmpty())
11         return false;
12       else {
13         boolean successfullyReachedGoal = false
14         for (Action action : applicableActions){
15           State nextState = state.apply(action);
16           successfullyReachedGoal = (recursiveCreateGraph(nextState) ||
                 successfullyReachedGoal);
17           for (ContextSignal contextSignal : allContextSignals){
18             if (contextSignal.isRelevantForActionAndState(action, state){
19               State stateAfterContextSignal =
                     state.applyContextSignal(action, contextSignal);
20               recursiveCreateGraph(stateAfterContextSignal);
21             }
22           }
23         }
24         return successfullyReachedGoal;
25       }
26     }
27   }
28 }
```

Listing 7.12 Pseudocode of the class Action

```
1  class Action {
2    List<BeliefStateTuple> preconditions
3    List<BeliefStateTuple> effects
4    boolean isApplicableInState(State previousState){
5      for (BeliefStateTuple preconditionTuple in preconditions) {
6        ContextTuple stateTuple = state.bst.find { e -> e.variable ==
             signalTuple.variable};
7        if (stateTuple != null){
8          ContextTuple intersectionTuple =
               preconditionTuple.intersect(stateTuple);
9          if (intersectionTuple == null)
10           return false;
11       } else return false;
12     }
13     return true;
14   }
15 }
```

Listing 7.13 Pseudocode of the class ContextSignal

```
1  class ContextSignal {
2    List<BeliefStateTuple> preconditions
3    List<BeliefStateTuple> effects
4    boolean isRelevantForActionAndState(Action a, State previousState){
5      boolean foundCommonContextVariable = false;
6      for (ContextTuple signalTuple in effects) {
7        ContextTuple actionTuple = a.preconditions.find { e -> e.variable ==
             signalTuple.variable};
8        if (actionTuple != null){
9          foundCommonContextVariable = true;
10         break;
11       }
12     }
13     if (!foundCommonContextVariable)
14       return false;
15     for (BeliefStateTuple signalTuple in preconditions) {
16       BeliefStateTuple actionTuple = a.preconditions.find { e -> e.variable
           == signalTuple.variable};
17       BeliefStateTuple stateTuple = state.bst.find { e -> e.variable ==
           signalTuple.variable};
18       if (actionTuple != null && stateTuple != null){
19         BeliefStateTuple intersectionTuple =
             signalTuple.intersect(actionTuple).intersect(stateTuple);
20         if (intersectionTuple == null)
21           return false;
22       }
23     }
24     return true;
25   }
26 }
```

Listing 7.14 Pseudocode of the class State

```
1  class State {
2    List<BeliefStateTuple> bst
3    List<Action> getApplicableActions(){
4      List<Action> applicableActions = new List Action();
5      for (Action action : allAction){
6        if (action.isApplicableInState(this))
7          applicableActions.add(action);
8      }
9      return applicableActions;
10   }
11   State apply(Action action){
12     State nextState = this.clone();
13     nextState.intersectBeliefStateTuples(action.preconditions);
14     nextState.insertOrReplaceBeliefStateTuples(action.effects);
15     return nextState;
16   }
17   State applyContextSignal(Action action, ContextSignal contextSignal){
18     State nextState = this.clone();
19     nextState.intersectBeliefStateTuples(action.preconditions);
20     nextState.insertAndUnifyBeliefStateTuples(action.effects);
21     nextState.intersectBeliefStateTuples(contextSignal.preconditions);
22     nextState.insertOrReplaceBeliefStateTuples(contextSignal.effects);
23     return nextState;
24   }
25 }
```

7.2.3 Mathematical Proofs of Key Properties

Theorem 1. The execution of the algorithm terminates.

> **Proof sketch.** Termination is shown by proving that all relevant sets are finite and the algorithm could not reach an endless loop. We will, in the following, show that the algorithm terminates.
>
> As the set A of process actions is finite, there are only finitely many actions that are applicable in a belief state bs. Further, there are only finitely many next belief states that are constructed by means of the regular state transition. In each of those newly constructed belief states then again finite actions are applicable. As the effects of actions are independent of the belief state the action was planned in, there can only arise finitely many belief states.
>
> Hence, it is sufficient to prove that a belief state could not be considered twice by means of our search algorithm (cf., due to a loop in the planning graph). This is ensured by the termination criteria of the algorithm. If an already considered belief state is reached, the forward search stops per definition.

By means of this criterion we ensure that a belief state is only considered once.

To prove that the planning of receive context actions terminates we need to consider the set of context signals *SIG*, which is finite as well. Thus, for each (of finitely many) combination of a belief state and a following action, only finite context signals could occur. Further, as a context signal (i.e., the effects of the according receive context action) is independent of the belief state and the action during which it occurs, there can only arise finitely many belief states by means of context signals. From these belief states the planning starts again, and during this newly planned new context signals could be relevant. As the termination criteria are considered here as well, we have proven that the planning of context-aware process models terminates. *q.e.d.*

Further, to ensure a comprehensive process model, we need to ensure that our approach considers all context signals and their according receive context actions (i.e., completeness).

Theorem 2. The approach is complete (i.e., it plans all receive context actions for all context signals, that could occur during planned actions).

Proof sketch. Assume there is a context signal *sig* that could occur and an according receive context action c_{sig} that is not planned. Following Definition 2.2.8 the planning graph has to contain a belief state bs followed by an action *a* satisfying $\forall t \in effects\left(c_{\text{sig}}\right) \exists u \in precond\left(a\right) | v\left(t\right) = v\left(u\right)) \wedge (\forall x \in precond\left(c_{\text{sig}}\right) \exists w \in bs \mid \mathrm{v}\left(w\right) = v\left(x\right) \wedge \left(r\left(x\right) \cap r\left(w\right) \neq \varnothing\right))$ so that sig could occur . As our algorithm retrieves all context signals and according receive context actions for all planed actions and *sig* could occur during the conduction of a, c_{sig} is planned and we have a contradiction. *q.e.d.*

Theorem 3. The approach is correct (i.e., it does not plan receive context actions according to context signals that could not occur)

Proof sketch. Let us assume a context signal *sig* that could not occur during any action exists and an according receive context action c_{sig} is planned by the algorithm. As there is a point at which c_{sig} could have been planned, there has to be an action *a* in a belief state *bs* so that Definition 2.2.8 is met. As this is also the condition that a context signal could occur our assumption is not true and thus the algorithm is correct. *q.e.d.*

Further, we evaluated the computational complexity of our approach, which means, we analyzed based on our nondeterministic context-aware state transition system the planning of receive context actions. The following two statements must both hold in order to plan a receive context action:

$\exists\, t \in \mathrm{eff}(c_{sig})$: $\exists u \in \mathrm{pre}(a)$, $v(t)=v(u)$ and

$\forall\, x \in \mathrm{pre}(c_{sig})\ \exists\, w \in bs : v(w) = v(x) \wedge ((\nexists\, y \in \mathrm{pre}(a) : v(y) = v(x) \wedge r(x) \cap r(w) \neq \varnothing) \vee (\exists\, y \in \mathrm{pre}(a) : v(y) = v(x) \wedge r(x) \cap r(y) \cap r(w) \neq \varnothing))$.

To ensure that 1) holds, at least one context variable must be present in both $\mathit{eff}(c_{sig})$ and $\mathit{pre}(a)$. Both sets must be compared pairwise and thus, the worst case here is that only the context variables in both sets compared at last are equal (i.e., $v(t)=v(u)$) and both sets do not overlap in any other context variable. Thus, the pairwise comparison of the context variables in those two sets results in a complexity of $O(|CT|^2)$, $O(|C|)$ and $O(|A|)$. For the second statement, $|BS|$ belief states, $|A|$ preconditions of actions and $|C|$ preconditions of receive context actions must be compared to each other and for each of these combinations two pairwise comparisons for each ($|BST|$) belief state tuple are required. Thus, the computational complexity of our approach is $O(|BST|^2)$, $O(|CT|)$, $O(|C|)$, $O(|BS|)$ and $O(|A|)$. To sum it up, in the worst-case-scenario, the algorithm is in quadratic time in the number of context tuples or belief state tuples, which means, the algorithm is computationally efficient (cf., Arora and Barak, 2009; Cobham, 1965).

7.3 Paper 3: The Cooperation of Multiple Actors within Process Models: An Automated Planning Approach

7.3.1 Pseudocode of the Main Primitives of our Algorithm

Listing 7.15 CreateWorldStates – Primitive *main*

```
1  SUB main
2    StatesThatLeadToGoal = {};
3    FOR actor ∈ ACTORS
4      bs = actor.initial_state
5      result = plan(bs)
6      IF result == true
7        StatesThatLeadToGoal.add(bs)
8  ENDSUB
```

Listing 7.16 CreateWorldStates – Primitive *plan*

```
1  SUB plan(bs)
2    result = false
3    FOR a ∈ ACTIONS
4      IF checkForApplicability(a in bs) == true
5        stateTransition(bs, a)
6      ELSE IF checkApplicAfterDisband(a in bs) == true
7        result = disband(bs, a)
8      ELSE IF checkApplicWithJoins(a in bs) == true
9        result = join(bs, a)
10   RETURN result
11 ENDSUB
```

Listing 7.17 CreateWorldStates – Primitive *stateTransition*

```
1  SUB stateTransition(bs, a)
2    result = false
3    bsnew = R(bs, a)
4    IF bsnew is goal
5      result = true
6    ELSE
7      result = plan(bsnew)
8    RETURN result
9  ENDSUB
```

Listing 7.18 CreateWorldStates – Primitive *join*

```
1  SUB join(bs, a)
2    saveForFutureJoins(bs, a)
3    JointStates = retrieveValidJointStates(bs)
4    joinPossible = false
5    FOR jointState ∈ JointStates
6      IF stateTransition(jointState, a) == true
7        constructJoinActions(jointState)
8      joinPossible = true
9    RETURN joinPossible
10 ENDSUB
```

Listing 7.19 CreateWorldStates – Primitive *disband*

```
1  SUB disband(bs, a)
2    DisbandedStates = retrieveValidDisbStates(bs)
3    disbandPossible = false
4    FOR disbandedState ∈ DisbandedStates
5     IF stateTransition(disbandedState, a) == true
6       constructSplitAction(disbandedState)
7       disbandPossible = true
8     ELSE IF checkApplicabilityWithJoin(a in disbandedState) == true
9       IF join(disbandedState, a) == true
10        constructSplitAction(disbandedState)
11        disbandPossible = true
12   RETURN disbandPossible
13 ENDSUB
```

7.3.2 Mathematical Evaluation

Theorem 1. The execution of the algorithm terminates.

> **Proof sketch.** Termination is shown by proving that only a finite number of iteration steps is performed, and that each iteration step of the algorithm terminates. Let *s=1,2,..* be the iteration steps and *S* be the set of all performed iteration steps.
>
> We first prove that $|S| < \infty$. Let $R(bs, a)$ be the transition function which, for an action a applicable in a belief state bs, provides the belief state resulting from the application of a in bs, and $R(bs, a)=\varnothing$ for an action a not applicable in bs. Let $\bigcup_{i \in ACTORS} \bigcup_{a \in ACTIONS} R(\text{Init}_i, a) =: bs_1$. Due to $|ACTIONS| < \infty$ and $|ACTORS| < \infty$, $|bs_1| < \infty$. Iteratively defining $bs_k := \bigcup_{bs \in bs_{k-1}} \bigcup_{a \in ACTIONS} R(bs, a)$ for each $k \in \mathbb{N}$, $k \geq 2$, it equivalently follows that $|bs_k| < \infty$ for each $k \in \mathbb{N}$ because of $|bs_{k-1}| < \infty$ and $|ACTIONS| < \infty$. There is a $l \in \mathbb{N}$ such that for all $bs \in bs_l$: $bs \in \bigcup_{k=1,\ldots,l-1} bs_k \cup$

$\bigcup_{i \in ACTORS} \text{Init}_i$, hence $|\bigcup_{k \in \mathbb{N}} bs_k| < \infty$. In other words, only a finite number of different belief states can be constructed based on the application of actions in *ACTIONS* to the initial states and the thereby constructed belief states (*). Additionally, join actions and split actions can be constructed during the course of the algorithm; let *C* be the set comprising all constructed join actions and *D* be the set comprising all constructed split actions. Due to $|ACTORS| < \infty$ and (*), $|C| < \infty$ and $|D| < \infty$. Thus, altogether, the number of actions considered and planned is finite because of $|ACTIONS|+|C|+|D| < \infty$. Following this, analogous to above, the number of different belief states constructed by the algorithm is finite (**).

If a belief state *bs* has already been considered in an earlier iteration step, the algorithm does not perform another iteration step for *bs*. Because of (**), there is a $t \in \mathbb{N}$ and an iteration step s_t in which all belief states have already been considered in the iteration steps $s_1, ..., s_{t-1}$. Hence, altogether, $|S| < \infty$.

An iteration step of the algorithm consists of the sub steps ***1a*)-*1c*)** and ***2a*)-*2e*)** described in the algorithm subsection. Step ***1a*)** terminates since only a finite number of actions needs to be checked for applicability (as $|ACTIONS| < \infty$), and each such check terminates as just a finite number of simple set comparisons is required (cf. Definition 2.3.11). Step ***1b*)** terminates because the criteria i. and ii. for splitting a belief state can be checked trivially and the termination of examining criterion iii. is equivalent to the termination of step ***1a*)**, which was already proved. Step ***1c*)** terminates obviously as only a finite number of simple set comparisons is necessary. Steps ***2a*)-*2e*)** need to be performed only a finite number of times in each $s \in S$ because of $|ACTIONS| < \infty$. Step ***2a*)** terminates also due to this reason and because of $|\mathit{eff}(a)| < \infty$ for each $a \in ACTIONS$. Step ***2b*)** terminates as the creation of a split action only requires a finite number of simple set operations and the subsequent belief state is constructed just like in step ***2a*)**. Step ***2c*)** is computationally trivial. Step ***2d*)** terminates because due to $|ACTIONS| < \infty$ and (**), only a finite number of combinations needs to be checked, and each check is equivalent to performing step ***1a*)**. Step ***2e*)** terminates due to the same reasons as step ***2b*)**. *q.e.d.*

Theorem 2. The algorithm constructs correct process models: All planned paths are feasible.

Proof sketch. To prove the correctness of the generated process models, it suffices to show that 1) the actions generated by the algorithm do not lead

to logical contradictions within the process model, and that 2) in no belief state an action which is not applicable can be planned by the algorithm. We start with 1). In step ***2b***) of the algorithm, split actions are generated; let *bs* be a belief state which is split into the belief states bs_1,... ,bs_n. Because of criteria i. and ii., $A(bs_i) \cap A(bs_j) = \varnothing$ for all $i, j \in \{1, \ldots, n\}$ such that $i \neq j$ and $\cup_i A(bs_i) = A(bs)$, which leads to $A(\text{bs}) = \dot{\cup}_i A(bs_i)$ (disjoint union). Hence, split actions do not lead to logical contradictions. The logical consistency of the join actions generated in step ***2e***) of the algorithm is ensured by the respective criteria i. and ii. which prevent logically contradictory belief states: i. excludes belief states which contain the same actors from joining and ii. guarantees that belief states which contain non-actor-specific variables with contradicting restrictions cannot be joined. In regard to 2), actions are planned in the steps ***2a***), ***2b***) and ***2e***) of the algorithm. Actions planned are either actions included in *ACTIONS* or are join/split actions generated by the algorithm. As all of these actions are actions in the sense of Definition 2.3.10 and Definition 2.3.11 is used for the applicability check, we do not differentiate between them further. The actions planned in the steps ***2b***) and ***2e***) are applicable by definition (as their preconditions and cardinality match the considered belief state) and in step ***2a***), an applicability check is performed before planning an action. *q.e.d.*

Theorem 3. The algorithm constructs complete process models: All feasible paths leading from an initial state to a goal state are being planned.

Proof sketch. It suffices to show that starting from a belief state *bs*, all actions *a* in *ACTIONS* that can possibly be applied as next action are planned by our algorithm. Let $R(bs, a)$ be the transition function which, for an action *a* applicable in a belief state *bs*, provides the belief state resulting from the application of *a* in *bs*, and $R(bs, a)=\varnothing$ for an action *a* not applicable in *bs.* There are four cases:

(1) *a* is applicable in *bs*

(2) *a* is applicable in $R(bs, d)$, where *d* is a split action

(3) *a* is applicable in $R(bs, c)$, where *c* is a join action

(4) *a* is applicable in $R(R(bs, d), c)$, where *d* is a disband action and *c* is a join action

Ad (1): The action *a* is identified in step ***1a***) and planned in step ***2a***).

Ad (2): The action d is identified in step ***1b***) and planned in step ***2b***); thereafter, the action a is planned accordingly in the belief state $R(bs, d)$ (cf. (1)).

Ad (3): In steps ***1c***), ***2c***) and ***2d***) possibilities for join actions are identified. In particular, the matching performed in step ***2d***) guarantees the consideration of all possible state-combinations that can be joined. Thus, c is planned in step ***2e***), which enables the subsequent planning of a in the belief state $R(bs, c)$ (cf. (1)).

Ad (4): The possibility is identified in step ***1b***) in conjunction with steps ***1c***), ***2c***) and ***2d***) and the actions are planned accordingly. *q.e.d.*

7.4 Paper 4: Adapting Process Models via an Automated Planning Approach

7.4.1 Evaluation of (E1) Correctness and Completeness

In the following, we focus on proving that the presented approach for the adaptation of process graphs fulfills the properties correctness, completeness and termination. In our approach, we state to "conduct planning steps". Here, we make use of existing techniques for the automated planning of process models (e.g., Bertoli et al., 2006; Heinrich et al., 2015b) and refer to their works for providing proofs for the fact that planning by conducting planning steps fulfills the properties correctness, completeness and termination.

Theorem 1. *The process graphs constructed by the approach are correct: Only feasible paths are contained in an adapted process graph.*

Proof. We distinguish the possible changes as described in the approach.

Updating the initial state. We show the feasibility of each path by using Definition 3.1.5 and, in particular, conditions i. to iii. To satisfy condition iii. of Definition 3.1.5, we examine all state transitions within the paths of the adapted process graph. In accordance with our approach, these transitions can be divided into transitions from old, updated and new states.

For each new and updated state the applicability of each following action in the adapted process model is verified and the following states are constructed by (partially) applying the transition function where needed, which leads to feasibility condition iii. being fulfilled by construction. Additionally, old states stem from the given process graph. Since only the initial state was updated, no further changes occur once an old state has been reached and thus the according subgraphs, which remain correct, are used for the adapted process graph. Hence, the feasibility condition iii. is fulfilled in these cases as well.

A path is completed as soon as we are certain to reach a goal state through an old, updated or new state. When no goal state is reached and no further actions can be applied, the current path is not considered in the adapted process graph. Thus, conditions i. and ii. of Definition 3.1.5 are fulfilled and the feasibility of every path in the adapted process graph is proven.

Adding a goal state. At first, we consider all paths feasible in the given process graph. For such a path $(bs_{\text{init}}, a_1, bs_2, \ldots, bs_{\text{k}})$, two cases may occur: In the

first case, bs_{init},$bs_{2,\ldots,}$ $bs_{\text{k-1}}$ do not meet the new goal state. Then, the path remains feasible as the conditions for a feasible path in Definition 3.1.5 are still fulfilled. In the second case, (at least) one of the belief states bs_{init},$bs_{2,\ldots,}bs_{\text{k-1}}$ meets the new goal state *goal*. In this case, the existing path is shortened until only one such belief state is contained and is the last belief state in the path. This shortened path is feasible as well: i. and ii. in Definition 3.1.5 are fulfilled by construction and iii. remains fulfilled. In addition to these kinds of paths, the adapted process graph may also contain new paths leading to *goal*. As we construct these paths by conducting planning steps (i.e., iteratively computing $app(bs)$ and applying the transition function for every belief state bs), the paths are always feasible.

Removing a goal state. We again consider all paths feasible in the given process graph. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ be such a path. If bs_{k} meets an element of $GOALS'$=$GOALS\backslash\{goal\}$, the path obviously remains feasible as the conditions in Definition 3.1.5 are still fulfilled. On the other hand, if the path had ended at *goal* it is extended until a belief state meeting a goal state from $GOALS'$ is reached. This is done by conducting planning steps until the extended path is feasible. If such an extension to a feasible path is not possible, the path, which formerly was feasible, is not considered in the adapted process graph. No further new paths are constructed. Hence, overall, only feasible paths are contained in the adapted process graph.

Updating a goal state. Strengthening update. Again, we consider all paths feasible in the given process graph. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ be such a path. If bs_{k} meets a goal state from $GOALS\backslash\{goal\}$, the path obviously remains feasible as the conditions in Definition 3.1.5 are still fulfilled. If the path had ended at *goal*, it is extended until a belief state meeting a goal state from $GOALS'$ is reached. This is done by conducting planning steps until the extended path is feasible. If such an extension to a feasible path is not possible, the path, which formerly was feasible, is not considered in the adapted process graph. No further new paths are constructed. Thus, similar to the case of removing a goal, only feasible paths are contained in the adapted process graph.

Updating a goal state. Weakening update. Yet again, we first consider all paths feasible in the given process graph. For such a path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$, two cases may occur: In the first case, $bs_{\text{init}},bs_{2,\ldots,}bs_{\text{k-1}}$ do not meet *goal'*. Then, the path remains feasible as the conditions for a feasible path in Definition 3.1.5 are still fulfilled. In the second case, (at least)

one of the belief states bs_{init},$bs_{2,\ldots,}$ $bs_{\text{k-1}}$ meets *goal'*. In this case, the existing path is shortened until only one such belief state is contained and is the last belief state in the path. This shortened path is feasible as well: i. and ii. in Definition 3.1.5 are fulfilled by construction and iii. remains fulfilled. In addition to these kinds of paths, the adapted process graph may also contain new paths leading to *goal'*. As we construct these paths by conducting planning steps, the paths are always feasible.

Adding an action. At first we note that all paths of the given process graph remain feasible when adding a to the set of actions since the conditions i. to iii. of Definition 3.1.5 are still fulfilled. In addition to that we try to reach goal states through new paths that include the action a. As we do this by conducting planning steps, the constructed paths are always ensured to be feasible.

Removing an action. When removing an action a from the set of actions, the adapted process graph contains the feasible paths of the given process graph which do not contain a. As every action in such a path still fulfills the applicability criterion and the last belief state is the only belief state that meets a goal state in such a path, these paths remain feasible (cf. Definition 3.1.5).

Updating an action. Strengthening update of the preconditions. The adapted process graph consists of paths retained from the given process graph and newly constructed paths. We start by considering all paths feasible in the given process graph. If such a path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ does not contain a at all, it obviously remains feasible as the path does not change at all and the conditions i.-iii. in Definition 3.1.5 are still fulfilled. Otherwise, there exists $i<k$ with $a = a_{\text{i}}$ and the applicability of a' in bs_{i} is checked. If $a' \in app(bs_{\text{i}})$ and $R(bs_{\text{i}},a) \neq R(bs_{\text{i}},a')$, one tries to construct a new path and proceeds as when treating the case of the initial state, for which correctness was already proven. Hence, all newly constructed paths are feasible. On the other hand, if $a' \in app(bs_{\text{i}})$ and $R(bs_{\text{i}},a) = R(bs_{\text{i}},a')$ *for all i, the conditions i.-iii. in Definition* 3.1.5 *remain fulfilled as all belief states in the path remain identical. Finally,* if $a' \notin app(bs_{\text{i}})$, the path is not considered in the adapted process graph. Hence, only feasible paths are retained from the given process graph.

Updating an action. Weakening update of the preconditions. In case of a weakening update of the preconditions of an action a resulting in the action a', all feasible paths of the given process graph which do not contain a are retained. Since these paths remain unchanged, they stay feasible. Furthermore, additional paths in the adapted process graph are retrieved by

conducting planning steps from belief states bs with $a \notin app(bs)$*; they are feasible by construction*. Additionally, for every belief state bs of the given process graph with $a \in app(bs)$ *and* $a' \in app(bs)$, we follow the approach for an update of the initial state. This approach, as seen above, leads to feasible paths.

Updating the effects. When updating the effects of an action a, one retains all paths $(bs_{\text{init}}, a_1, bs_2, \ldots, bs_{\text{k}})$ which do not contain a at all and obviously remain feasible as they do not change at all. Otherwise, for $a = a_{\text{i}}$ $(i<k)$, if $R(bs_{\text{i}}, a) \neq R(bs_{\text{i}}, a')$, one tries to construct a new path and proceeds as when treating the case of the initial state, for which correctness was already proven.

Thus, Theorem 1 is shown for each case and hence proven. *q.e.d.*

Theorem 2. *The process graphs constructed by the approach are complete: All feasible paths are contained in an adapted process graph.*

Proof. We distinguish the possible changes as described in the approach.

Updating the initial state. According to Definition 3.1.5, each feasible path has to start with the initial state and each following action has to be applicable in its preceding state and has to lead to a goal state (in the sense that the conduction of this action and potentially subsequent actions results in a goal state). As in the previous theorem, we therefore examine all state transitions from old, updated and new states in order to show that the adapted process graph contains every path of this nature. Let bs be a belief state.

In case of bs being an old state, we retain the subgraph from the given process model which starts with bs. As the given process graph is complete, this subgraph contains all actions from $app(bs)$ that lead to a goal state.

Let bs be an updated or new state. If bs itself does not meet a goal state, each action $a \in app(bs)$ is checked in regard to whether it leads to a goal state: For each such action a all possible subsequent actions and states are retrieved by applying the transition function until a goal state is reached, an old state is reached or no further action can be applied. In the first two cases, a indeed leads to (at least) one goal state and consequently the sequence bs, a is part of a feasible path. In the last case, on the contrary, every path containing the sequence bs, a is not considered in the adapted process graph as a does not lead to a goal state.

To sum up, beginning with bs_{init} for each old, updated or new state we either examine all applicable actions and thereafter retain those leading to a goal state or we retain a subgraph of the given process graph which contains all

applicable actions that lead to a goal state. Thus, the adapted process graph contains all feasible paths.

Adding a goal state. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ be any feasible path after the addition of *goal* to the set of goal belief states *GOALS* such that *GOALS'*=*GOALS*∪{*goal*}. We need to show that $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ is indeed contained in the adapted process graph. As $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ is feasible, according to Definition 3.1.5 i., bs_k meets one of the goal states from *GOALS'*=*GOALS*∪{*goal*}. In the first case, let bs_k meet one of the goal states from *GOALS*. In this case, $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ was feasible in the given process graph and - since $bs_{\text{init}},bs_2,bs_{k-1}$ do not meet a goal state from *GOALS'* because of Definition 3.1.5 ii. - is retained from the given process graph and thus contained in the adapted process graph. In the second case, bs_k meets *goal*. It is then possible that $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ was part of a feasible path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k,\ldots,bs_m)$ with $m>k$ in the given process graph. Such a path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k,\ldots,bs_m)$ is, according to the approach, shortened to $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ and then retained. Hence, $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ is contained in the adapted process graph. If $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ was not part of a feasible path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k,\ldots,bs_m)$ in the given process graph, (at least) one of the actions $a_1,a_2,\ldots$ was not planned in the given process graph in bs_{init}, resp. $bs_1,\ldots$ For such actions, planning steps are conducted, which lead to the inclusion of $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ into the adapted process graph. Thus, overall, it is guaranteed that $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ is contained in the adapted process graph.

Removing a goal state. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ be any feasible path after the removal of *goal* from the set of goal belief states *GOALS* such that *GOALS'*=*GOALS*\{*goal*}. If $bs_{\text{init}},bs_2,\ldots,bs_{k-1}$ do not meet *goal*, $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ was a feasible path in the given process graph (cf. conditions i.-iii. in Definition 3.1.5), which is retained and hence contained in the adapted process graph. Otherwise, let bs_m ($m<k$) be the first belief state that meets *goal*. Then, $(bs_{\text{init}},a_1,bs_2,\ldots,bs_m)$ was a feasible path in the given process graph which, by according to our approach, is extended to $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ – and possibly other additional feasible paths – in the adapted process graph.

Updating a goal state. Strengthening update. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ be any feasible path after the update of a goal state *goal* to *goal'*. If $bs_{\text{init}},bs_2,\ldots,bs_{k-1}$ do not meet *goal*, $(bs_{\text{init}},a_1,bs_2,\ldots,bs_k)$ was a feasible path in the given process graph (cf. conditions i.-iii. in Definition 3.1.5), which is re-

tained and hence contained in the adapted process graph. Otherwise, let bs_m be the first such belief state. Then, $(bs_{init},a_1,bs_2,\ldots,bs_m)$ was a feasible path in the given process graph which, according to our approach, is extended to $(bs_{init},a_1,bs_2,\ldots,bs_k)$ – and possibly other additional feasible paths – in the adapted process graph.

Updating a goal state. Weakening update. Let $(bs_{init},a_1,bs_2,\ldots,bs_k)$ be any feasible path after the update of a goal state *goal* to *goal'*. We need to show that $(bs_{init},a_1,bs_2,\ldots,bs_k)$ is indeed contained in the adapted process graph. As $(bs_{init,}a_1,bs_2,\ldots,bs_k)$ is feasible, according to Definition 3.1.5 i., bs_k meets one of the goal states from $GOALS'=(GOALS\backslash\{goal\})\cup\{goal'\})$. In the first case, let bs_k meet one of the goal states from $GOALS\backslash\{goal\}$. In this case, $(bs_{init},a_1,bs_2,\ldots,bs_k)$ was feasible in the given process graph and - since $bs_{init},bs_2,bs_{k\text{-}1}$ do not meet a goal state from *GOALS'* because of Definition 3.1.5 ii. - is retained from the given process graph and thus contained in the adapted process graph. In the second case, bs_k meets *goal'*. It is then possible that $(bs_{init},a_1,bs_2,\ldots,bs_k)$ was part of a feasible path $(bs_{init},a_1,bs_2,\ldots,bs_k,\ldots,bs_m)$ with $m\geq k$ in the given process graph. Such a path $(bs_{init},a_1,bs_2,\ldots,bs_k,\ldots,bs_m)$ is shortened to $(bs_{init},a_1,bs_2,\ldots,bs_k)$ and then retained. Hence, $(bs_{init},a_1,bs_2,\ldots,bs_k)$ is contained in the adapted process graph. If $(bs_{init},a_1,bs_2,\ldots,bs_k)$ was not part of a feasible path $(bs_{init},a_1,bs_2,\ldots,bs_k,\ldots,bs_m)$ in the given process graph, (at least) one of the actions $a_1,a_2,\ldots$ was not planned in the given process graph in bs_{init}, resp. $bs_2,\ldots$ For such actions, planning steps are conducted, which lead to the inclusion of $(bs_{init},a_1,bs_2,\ldots,bs_k)$ into the adapted process graph. Thus, overall, it is guaranteed that $(bs_{init},a_1,bs_2,\ldots,bs_k)$ is contained in the adapted process graph.

Adding an action. Let $(bs_{init},a_1,bs_2,\ldots,bs_k)$ be any feasible path after the addition of a to the set of actions. If a is not among the actions $a_1,a_2,\ldots$ of the aforementioned path, this feasible path is contained in the given process graph. As our approach retains all paths from the given process graph, this path is contained in the adapted process graph as well. Now let a be contained in the actions $a_1,a_2,\ldots$ of the selected feasible path. Then there is a belief state bs preceding (the first occurrence of) a in this path with $a\in app(bs)$. As elaborated in the design of our approach, we examine the path $(bs_{init},a_1,bs_2,\ldots,bs)$ as it ends with a state in which a is applicable. From here, we conduct planning steps to determine and retain all feasible paths

that extend $(bs_{\text{init}},a_1,bs_2,\ldots,bs)$ and hence $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ is contained in the adapted process graph.

Removing an action. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ be any feasible path after the removal of a from the set of actions A. As, in regard to Definition 3.1.5, actions must be part of the set of actions to be contained in a path, this path does not contain a. It is contained in the given process graph and, as such a path is not changed at all, retained in the adapted process graph according to our approach.

Updating an action. Strengthening update of the preconditions. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ be any feasible path after a strengthening update of the preconditions of an action a resulting in the action a'. If this path does not contain a', it remains unchanged, is retained from the given process graph and hence contained in the adapted process graph. If, however, the path contains a' at some place a_{i} (let i be the smallest index such that $a'=a_{\text{i}}$), a part of the considered path, more precisely $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{i}})$, is contained in the given process graph. From here, we proceed as in the case of updating the initial state and retrieve all feasible paths from bs_{i}. As this was proven to be complete, the feasible path $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ is retrieved as well.

Updating an action. Weakening update of the preconditions. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ be any feasible path after a weakening update of the preconditions of an action a resulting in the action a'. If this path does not contain a', it remains unchanged, is retained from the given process graph and hence contained in the adapted process graph. If, however, the path contains a' at some place a_{i} (let i be the smallest index such that $a'=a_{\text{i}}$), it is either retrieved by conducting planning steps from a belief state bs_{j} with $j\leq i$ of the given process graph or by following the approach for the initial state (which, as seen above, is complete).

Updating the effects. Let $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ be any feasible path after updating the effects of an action a resulting in the action a'. Again, if this path does not contain the updated action, the path is contained in the given process graph and, according to the approach, retained. Otherwise, there exists an index $i<k$ with $a'=a_{\text{i}}$ and $a'\neq a_{\text{j}}$ for all $j<i$. Here, we treat $R(bs_{\text{i}},a')$ as the update of $R(bs_{\text{i}},a)$ *and follow the approach for an update of the initial state which retrieves all feasible paths* (*seen in the proof above*) *starting with* $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{i}},a',R(bs_{\text{i}},a'))$, $(bs_{\text{init}},a_1,bs_2,\ldots,bs_{\text{k}})$ being *amongst them*.

Thus, Theorem 2 is shown for each case and hence proven. *q.e.d.*

Theorem 3. *The approach terminates.*

Proof. To show that the algorithm terminates, we distinguish the possible changes as described in the approach.

Updating the initial state. At first, we will show that the traversal of each belief state terminates. The traversal of an old state immediately terminates as we retain the corresponding subgraph. Traversing an updated state *bs*, we first note that the computation of the set $app(bs)$ comes to an end and that the set $app(bs)$ is finite. This stems from the fact that checking the applicability of an action a (i.e., $\forall w{\in}pre(a)\ \exists u{\in}bs:\ v(w){=}v(u) \wedge r(w){\cap}r(u) \neq \varnothing$) is a comparison of the restrictions of a finite number of belief state tuples and hence terminates and the fact that we are provided with a finite set of actions. As a next step, the finite set of all following belief states is retrieved by executing the state transition function on each action in $app(bs)$ which terminates as an operation on finite sets of belief state tuples. To check which of these belief states are old, new or updated, we need to compare each one of them with each belief state of the given process graph, which is a finite graph. Thus, handling an updated state terminates. Dealing with a new belief state *bs* terminates in a similar way as we again have to compute $app(bs)$, the belief states following *bs* and classify them as old, new or updated. To sum up, each traversal step terminates. Hence, it suffices to show that the number of traversal steps is finite which follows from the fact that the number of distinct belief states one can reach from an initial state by combining and conducting actions from a finite set of actions is finite.

Adding a goal state. The case of adding a goal state *goal* to the set of goal states *GOALS* is handled by a depth-first search through the belief states of the given process graph. Since the number of these belief states is finite (cf. Definition 3.1.5), it suffices to show that the traversal of each belief state terminates. Thus, we examine the traversal of a belief state *bs*. As a first step, we check whether *bs* meets *goal* (i.e., $\forall p{\in}goal$: $\exists p'{\in}bs$, $v(p){=}v(p')$, $r(p'){\subset}r(p)$), which is a comparison of the restrictions of a finite number of belief state tuples and hence terminates. If *bs* indeed meets *goal*, the traversal of *bs* ends. Otherwise, we try to reach a belief state meeting *goal* by conducting planning steps, which terminates as well.

Removing a goal state. For the finite number of paths of the given process graph which end at a belief state meeting *goal*, it is checked whether they can be extended so that they lead to one of the remaining goal states. This is

done by conducting planning steps for a finite number of belief states, which terminates.

Updating a goal state. Strengthening update. For the finite number of paths of the given process graph which end at a belief state meeting *goal*, it is checked whether they can be extended so that they lead to one of goal states from $GOALS' = (GOALS \setminus \{goal\}) \cup \{goal'\}$. This is done by conducting planning steps for a finite number of belief states, which terminates.

Updating a goal state. Weakening update. The case of updating a goal state *goal* to a goal state *goal'* which weakens the conditions of *goal* is handled by a depth-first search through the belief states of the given process graph. Since the number of these belief states is finite (cf. Definition 3.1.5), it suffices to show that the traversal of each belief state terminates. Thus, we examine the traversal of a belief state *bs*. As a first step, we check whether *bs* meets *goal'* (i.e., $\forall p \in goal' \colon \exists p' \in bs,\ v(p) = v(p'),\ r(p') \subset r(p)$) which is a comparison of the restrictions of a finite number of belief state tuples and hence terminates. If *bs* indeed meets *goal'*, the traversal of *bs* ends. Otherwise we try to reach a belief state meeting *goal'* by conducting planning steps, which terminates as well.

Adding an action. Adding an action *a* to the set of actions is handled by a depth-first search through the belief states of the given process graph. From each such belief state, planning steps are conducted in order to find belief states in which *a* is applicable and hence new feasible paths can possibly be constructed. As the conduction of planning steps terminates and the given process graph has a finite number of belief states (cf. Definition 3.1.5), the depth-first search terminates as well.

Removing an action. When removing an action *a* from A so that $A' = A \setminus \{a\}$, the finite set of all feasible paths of the given process graph is traversed. It is checked whether such a feasible path contains the action *a* in order to determine whether this path is retained. These checks terminate as each feasible path contains only a finite number of actions and thus our approach terminates.

Updating an action. Strengthening update of the preconditions. Again, all feasible paths of the given process graph are traversed in order to check whether in the belief states in which *a* was applicable, the updated action *a'* is applicable as well. If this is not the case, the path is not considered in the adapted process graph and the traversal of this path ends. On the

other hand, if a' is applicable in a belief state bs with $a \in app(bs)$, it is checked whether $R(bs,a)$ and $R(bs,a')$ coincide, which requires a finite amount of set comparisons and hence terminates. Finally, if these belief states do not coincide, we follow the approach for updating the initial state, which, as seen above, terminates. As the given process graph contains a finite number of feasible paths, this means that our approach addressing the strengthening update of the preconditions of an action terminates.

Updating an action. Weakening update of the preconditions. This case is handled by a depth-first search through the belief states of the given process graph. From each belief state, planning steps are conducted in order to find belief states *bs with* $a \notin app(bs)$ *and* $a' \in app(bs)$ and possibly construct new feasible paths containing a', which terminates. Additionally, there may be belief states *bs in which both a and a' are applicable. In this case*, it is checked whether $R(bs,a)$ and $R(bs,a')$ coincide, which requires a finite amount of set comparisons and hence terminates. *If these belief states do not coincide, we apply our approach for updating the initial state, which, as seen above, terminates.* As the given process graph has a finite number of belief states (cf. Definition 3.1.5), the depth-first search terminates as well.

Updating the effects. When updating the effects of an action a resulting in the action a', we traverse each belief state of the given process graph in which *a is applicable. As by Definition* 3.1.5 *the given process graph contains a finite number of belief states, the set of such belief states is finite as well. For each such belief state bs, we treat* $R(bs,a')$ *as the update of* $R(bs,a)$ *and apply our approach for updating the initial state, which terminates as seen above.*

Thus, Theorem 3 is shown for each case and hence proven. *q.e.d.*

7.4.2 Pseudocode of the Presented Approach

Listing 7.20 Pseudocode of the procedure updateinit

```
1 def updateinit(updated_inital_state, original_inital_state):
2   handleUpdatedState(updated_inital_state, original_inital_state)
```

Listing 7.21 Pseudocode of the procedure handleUpdatedState

```
1  def handleUpdatedState(updated_state, original_state):
2    if checkForGoal(updated_state):
3      return
4    old_actions = originalModel.getFollowingActions(updated_state)
5    new_actions = actionLibrary - old_actions
6    for action in old_actions:
7      if (action.preconditions.containsVariable(updated_belief_state_tuple)
           and isApplicable(action, updated_belief_state_tuple)) or not
           action.preconditions.containsVariable(updated_belief_state_tuple):
8        old_following_state = originalModel.getFollowingState(original_state,
             action)
9        new_following_state =
             old_following_state.update(updated_belief_state_tuple)
10       adaptedModel.addTransition(updated_state, action, new_following_state)
11       if originalModel.contains(new_following_state):
12         adaptedModel.addAll(originalModel.getSubgraphFromState(
               new_following_state))
13       else:
14         handleUpdatedState(new_following_state)
15   for action in new_actions:
16     if action.preconditions.containsVariable(updated_belief_state_tuple) and
         isApplicable(action, updated_belief_state_tuple):
17       following_state = apply(action, updated_state)
18       adaptedModel.addTransition(updated_state, action, following_state)
19       if originalModel.contains(following_state):
20         adaptedModel.addAll(originalModel.getSubgraphFromState(following_state))
21       elif isUpdatedState(following_state):
22         handleUpdatedState(following_state)
23       else:
24         handleNewState(following_state)
```

Listing 7.22 Pseudocode of the procedure handleNewState

```
1  def handleNewState(new_state):
2    if checkForGoal(new_state):
3      return
4    following_states = planStateTransitions(getApplicableActions(new_state),
       new_state)
5    for following_state in following_states:
6      if originalModel.contains(following_state):
7        adaptedModel.addAll(originalModel.getSubgraphFromState(following_state))
8      elif isUpdatedState(following_state):
9        handleUpdatedState(following_state)
10     else:
11       handleNewState(following_state)
```

Listing 7.23 Pseudocode of the procedure addGoal

```
1  def addGoal(new_goal_state):
2    adaptedModel.addGoal(new_goal_state)
3    for state in originalModel.states:
4      if checkForParticularGoal(state, new_goal_state):
5        adaptedModel.removeTransitions(originalModel.getSubgraphFromState(state))
6    for state in unplannedStates(originalModel):
7      if checkForParticularGoal(state, new_goal_state):
8        adaptedModel.addTransitions(originalModel.getTransitionsFromTo(
             inital_state, state))
```

Listing 7.24 Pseudocode of the procedure removeGoal

```
1  def removeGoal(old_goal_state):
2    adaptedModel.removeGoal(old_goal_state)
3    for state in pathEndingStates:
4      if checkForParticularGoal(state, old_goal_state):
5        planSubGraphFromState(adaptedModel, state)
6    adaptedModel.removeTransitionsNotLeadingToGoalStates()
```

Listing 7.25 Pseudocode of the procedure updateGoal

```
1  def updateGoal(updated_goal_state, original_goal_state):
2    if isStrengtheningUpdate(updated_goal_state, original_goal_state):
3      adaptedModel.addGoal(new_goal_state)
4      removeGoal(original_goal_state)
5    elif isWeakeningUpdate(updated_goal_state, original_goal_state):
6      adaptedModel.removeGoal(original_goal_state)
7      addGoal(updated_goal_state)
8    else:
9      addGoal(updated_goal_state)
10     removeGoal(original_goal_state)
```

Listing 7.26 Pseudocode of the procedure addAction

```
1  def addAction(new_action):
2    actionLibrary.add(new_action)
3    for state in originalModel.states:
4      if isApplicable(new_action, state):
5        following_state = apply(new_action, state)
6        adaptedModel.addTransition(state, new_action, following_state)
7        planSubGraphFromState(adaptedModel, following_state)
8    for state in unplannedStates(originalModel):
9      if isApplicable(new_action, state):
10       adaptedModel.addTransitions(originalModel.getTransitionsFromTo(
             inital_state, state))
11       following_state = apply(new_action, state)
12       adaptedModel.addTransition(state, new_action, following_state)
13       planSubGraphFromState(adaptedModel, following_state)
```

Listing 7.27 Pseudocode of the procedure removeAction

```
1  def removeAction(old_action):
2    actionLibrary.remove(old_action)
3    for transition in adaptedModel.stateTransitions:
4      if old_action in transition:
5        adaptedModel.removeTransition(transition)
6    adaptedModel.removeTransitionsNotLeadingToGoalStates()
```

Listing 7.28 Pseudocode of the procedure updateAction

```
1  def updateAction(updated_action, original_action):
2    actionLibrary.remove(original_action)
3    actionLibrary.add(updated_action)
4    if updated_action.preconditions != original_action.preconditions:
5      if isStrengtheningUpdate(updated_action.preconditions,
           original_action.preconditions):
6        strengtheningUpdatePreconditions(adaptedModel, updated_action,
             original_action)
7      elif isWeakeningUpdate(updated_action.preconditions,
           original_action.preconditions):
8        weakeningUpdatePreconditions(adaptedModel, updated_action,
             original_action)
9      else:
10       weakeningUpdatePreconditions(adaptedModel, updated_action,
             original_action)
11       strengtheningUpdatePreconditions(adaptedModel, updated_action,
             original_action)
12   if updated_action.effects != original_action.effects:
13     for transition in adaptedModel.stateTransitions:
14       if original_action in transition:
15         replaceTransitionAndUpdate(transition, updated_action)
```

Listing 7.29 Pseudocode of the procedure strengtheningUpdatePreconditions

```
1  def strengtheningUpdatePreconditions(adaptedModel, updated_action,
       original_action):
2    for transition in adaptedModel.stateTransitions:
3      if original_action in transition:
4        if isApplicable(updated_action, transition.fromState()):
5          replaceTransitionAndUpdate(transition, updated_action)
6        else:
7          adaptedModel.removeTransition(transition)
8          adaptedModel.removeTransitionsNotLeadingToGoalStates()
```

Listing 7.30 Pseudocode of the procedure weakeningUpdatePreconditions

```
1  def weakeningUpdatePreconditions(adaptedModel, updated_action,
       original_action):
2    for state in originalModel.states:
3      if isApplicable(original_action, state):
4        transition = adaptedModel.findTransition(state, original_action)
5        replaceTransitionAndUpdate(transition, updated_action)
6      elif isApplicable(updated_action, state):
7        following_state = apply(updated_action, state)
8        adaptedModel.addTransition(state, updated_action, following_state)
9        planSubGraphFromState(adaptedModel, following_state)
10   for state in unplannedStates(originalModel):
11     if isApplicable(updated_action, state):
12       adaptedModel.addTransitions(originalModel.getTransitionsFromTo(
             inital_state, state))
13       following_state = apply(updated_action, state)
14       adaptedModel.addTransition(state, updated_action, following_state)
15       planSubGraphFromState(adaptedModel, following_state)
```

Listing 7.31 Pseudocode of the procedure replaceTransitionAndUpdate

```
1  def replaceTransitionAndUpdate(transition, updated_action):
2    adaptedModel.removeTransition(transition)
3    old_following_state = transition.fromState()
4    updated_belief_state_tuple = applyForUpdatedBeliefState(updated_action,
         old_following_state)
5    new_following_state =
         old_following_state.update(updated_belief_state_tuple)
6    adaptedModel.addTransition(transition.fromState(), updated_action,
         new_following_state)
7    handleUpdatedState(new_following_state)
```

k	Number of belief states that are planned or otherwise known during planning
k_{old}	Number of belief states in the original process graph
k_{new}	Number of belief states in the adapted process graph
$k_{\text{unplanned}}$	Number of belief states which are reachable from the initial state, but not planned in the process graph (i.e., they do not lead to a goal state)
n	Number of all actions
m	Number of all belief state variables
g	Number of goal states

Table 7.1: Notation

7.4.3 Evaluation of Computational Complexity

In the following, we outline the differences in complexity between the presented adaptation of a process graph and planning the adapted process graph from scratch. To this end, we use the notation found in Table 7.1. If necessary, further notation is provided for each adaptation case.

Updating the initial state. Let $n_{\text{old}}=|app(bs_{\text{old}})|$ be the number of actions applicable in an old belief state. Evaluating the applicability in such a belief state can be done just for the updated belief state tuple. The same holds for the application of the transition function. Hence, these two steps require no more than $2+n_{\text{old}}$ comparisons using the presented approach versus $m*(2+n_{\text{old}})$ comparisons when planning from scratch. Having determined the following belief states to each applicable action, the effort of checking whether these states are already planned or meeting a goal state condition is the same for both approaches with $(k+g)*m$ comparisons. Please note that for every belief state, which is contained in the original process model the entire subgraph is adopted by our adaptation approach which takes (virtually) no effort. In contrast, when planning from scratch in a worst case scenario every combination of actions is feasible and thus $n!$ planning steps are required with each planning step consisting of $(k+g+3)*m*n$ comparisons.

Adding a goal state. As shown in Section 4.2.1, adding a goal state is addressed in two possible ways. Firstly, paths are shortened by checking each belief state of the existing process graph for the goal condition of the added goal state. Once this check yields true, removing all following edges and nodes is of insignificant computational cost which leads to a total of $k_{\text{old}}*m$ comparisons needed versus at least k_{new} planning steps with $(k+g+3)*m$ comparisons when planning from scratch. Secondly, new feasi-

ble paths are planned which lead to the added goal state by conducting $k_{\text{unplanned}}$ planning steps (versus k_{old}+$k_{\text{unplanned}}$ planning steps when planning from scratch). The reduction of complexity is even more substantial if all reachable believe states have been stored in the course of computing the original process graph. In this scenario, the presented approach does not need to execute planning steps. Instead, all reachable states have to be checked regarding the added goal state condition which in total requires (k_{goal}+$k_{\text{unplanned}}$)*m comparisons.

Removing a goal state. Let k_{goal} be the number of belief states, which meet the goal condition of the removed goal state. The task at hand is to search for new paths to the remaining goal states beginning from the belief states meeting the removed goal state condition. Thus, the presented approach reuses and modifies the original process graph where necessary by conducting planning from the aforementioned belief states. This leads to k_{goal}*g*m comparisons and $k_{\text{unplanned}}$ planning steps when adapting the process graph compared to k_{old}+$k_{\text{unplanned}}$ planning steps when planning from scratch. Again, if all reachable believe states are accessible, the complexity can be reduced to (k_{goal}+$k_{\text{unplanned}}$)*g*m comparisons to check all reachable states regarding the remaining goal state conditions.

Updating a goal state. In the worst possible case, the update of a goal state is addressed by the two steps above (i.e., adding the updated goal state and thereafter removing the obsolete goal state). With this in mind, the computational effort of these two steps can be added and compared to planning the updated process graph from scratch leading to k_{old}*m+k_{goal}*g*m+$k_{\text{unplanned}}$*(k+g+3)*m*n comparisons versus at least k_{new} planning steps with (k+g+3)*m*n comparisons.

Adding an action. Again, the presented approach fully makes use of the original process graph and tries to plan new paths by applying the added action where possible. Contrarily, planning the original process graph from scratch amounts to at least k_{old} planning steps, each containing (k+g+3)*m*(n-1) comparisons. In both approaches the applicability of the added action is checked for each state of the original process graph which accounts for m*k_{old} comparisons. If applicable, the state transition (2*m comparisons) as well as further planning steps ((k+g+3)*m*n comparisons each) are computed. As the added action can be applicable in reachable belief states, which are not contained in the original process graph, the computation of these belief states can be skipped when adapting the process graph. Here, only the applicability of the added action is determined, resulting in m*$k_{\text{unplanned}}$ comparisons opposed to $k_{\text{unplanned}}$ planning steps with (k+g+3)*m*n comparisons.

Removing an action. The presented approach identifies and deletes all paths that contain the removed action, which has no significant computational complexity. How-

ever, as seen above, planning the adapted graph from scratch requires at least k_{new} planning steps.

Updating an action. Updating the effects of an action does not affect its applicability. Instead, each following belief state has to be updated regarding the updated belief state tuple, which requires 2 comparisons. Afterwards, each updated state is handled in the same way as an updated initial state. Hence, we refer to the discussion above. When conducting a weakening update of the preconditions, the presented approach proceeds is similar to adding an action. Additionally, belief states, which follow the updated action in the original process graph, might be updated and treated as above. Analogously, a strengthening update of the preconditions leads to the removal of each path containing the updated action if it is not applicable. Otherwise, the following belief state is updated and handled accordingly.

Overall, the complexity analysis shows that our approach provides considerable advantages regarding computational complexity compared to planning process graphs from scratch.