Jens Gulden · Iris Reinhartz-Berger
Rainer Schmidt · Sérgio Guerreiro
Wided Guédria · Palash Bera (Eds.)

LNBIP 318

Enterprise, Business-Process and Information Systems Modeling

19th International Conference, BPMDS 2018
23rd International Conference, EMMSAD 2018, Held at CAiSE 2018
Tallinn, Estonia, June 11–12, 2018, Proceedings

Springer

Jens Gulden · Iris Reinhartz-Berger
Rainer Schmidt · Sérgio Guerreiro
Wided Guédria · Palash Bera (Eds.)

Enterprise, Business-Process and Information Systems Modeling

19th International Conference, BPMDS 2018
23rd International Conference, EMMSAD 2018
Held at CAiSE 2018, Tallinn, Estonia, June 11–12, 2018
Proceedings

Springer

Editors
Jens Gulden
University of Duisburg-Essen
Essen
Germany

Iris Reinhartz-Berger
University of Haifa
Haifa
Israel

Rainer Schmidt
Munich University of Applied Sciences
Munich
Germany

Sérgio Guerreiro
INESC-ID
University of Lisbon
Lisbon
Portugal

Wided Guédria
Luxembourg Institute of Science
and Technology
Esch-sur-Alzette
Luxembourg

Palash Bera
Saint Louis University
St. Louis, MO
USA

ISSN 1865-1348 ISSN 1865-1356 (electronic)
Lecture Notes in Business Information Processing
ISBN 978-3-319-91703-0 ISBN 978-3-319-91704-7 (eBook)
https://doi.org/10.1007/978-3-319-91704-7

Library of Congress Control Number: 2018944292

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG
part of Springer Nature
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This book contains the proceedings of two long-running events held along with the CAiSE conferences relating to the areas of enterprise, business process and information systems modeling: the 19th International Conference on Business Process Modeling, Development and Support (BPMDS 2018) and the 23rd International Conference on Evaluation and Modeling Methods for Systems Analysis and Development (EMMSAD 2018). The two working conferences are introduced below.

BPMDS 2018

The topics addressed by the BPMDS series, in conjunction with CAiSE (Conference on Advanced Information Systems Engineering), are focused on business processes and their IT support. This is one of the keystones of information systems theory beyond short-lived fashions. The continued interest in this topic on behalf of the information systems community is reflected by the success of the past BPMDS events, and their promotion from a workshop to a working conference.

The BPMDS series produced 18 events from 1998 to 2017. From 2011, BPMDS became a two-day working conference attached to CAiSE. The basic principles of the BPMDS series are:

1. BPMDS serves as a meeting place for researchers and practitioners in the areas of business development and business applications (software) development.
2. The aim of the event is mainly discussions, rather than presentations.
3. Each event has a theme that is mandatory for idea papers.
4. Each event's results are, usually, published in a special issue of an international journal.

The goals, format, and history of BPMDS can be found on the website: http://www.bpmds.org/.

BPMDS solicits papers related to business process modeling, development, and support (BPMDS) using quality, relevance, originality, and applicability as main selection criteria. As a working conference, BPMDS 2018 aimed to attract *full research papers* describing mature research, *experience reports* related to using BPMDS in practice, and visionary *idea papers*. To encourage new and emerging challenges and research directions in the area of business process modeling, development and support, BPMDS has a unique focus theme every year. Papers submitted as idea papers are required to be of relevance to the focus theme, thus providing a mass of new ideas around a relatively narrow but emerging research area. Full research papers and experience reports do not necessarily need to be directly connected to this theme.

The focus theme for BPMDS 2018 idea papers was "Ecosystem-Aware Business Process Modeling, Development, and Support." For the 19th edition of the BPMDS conference, we invited the interested authors to engage during the two days of BPMDS 2018 in Tallinn, and to take part in a deep discussion with all participants about the challenges of business transformation in the digitally connected world and the ways *business process modeling, development, and support* may provide capabilities to deal with these challenges. The challenges result from, among others, the impacts of the ubiquity of the actors, social networks, and new business models as well as the co-existence of flexibility, exception handling, context awareness, and personalization requirements together with other compliance and quality requirements.

Practitioners are producing business process models, researchers are studying and producing business process models, and are also producing new modeling languages when they consider that existing ones are not sufficient. What is beyond? Which kind of analyses can we make using these process models? How can we complete and enhance these process models with annotations, with data coming from everywhere out of the immediate process environment? How can the understanding we gain by working on these models in a sandbox help or facilitate the undergoing business transformation?

BPMDS 2018 received 29 submissions from 23 countries (Austria, Denmark, Egypt, Estonia, France, Germany, Greece, Israel, Italy, Latvia, Libya, The Netherlands, New Zealand, Norway, Pakistan, Poland, Russia, Saudi Arabia, Slovenia, Spain, Sweden, Switzerland, and Tunisia). Each paper received at least three reviews from the members of the international Program Committee. Eventually, 13 high-quality papers were selected, among them 11 research papers, one experience report, and one idea paper. The accepted papers cover a wide spectrum of issues related to business process development, modeling, and support. They are organized under the following section headings:

- Context-Awareness in Business Processes
- Automatic Analysis of Business Processes
- Advanced Approaches for Business Process Modeling
- Evaluation of Business Process Modeling Techniques
- An Experience Report on Modeling Collaborative Processes

We wish to thank all the people who submitted papers to BPMDS 2018 for having shared their work with us, as well as the members of the BPMDS 2018 Program Committee, who made a remarkable effort in reviewing submissions. We also thank the organizers of CAiSE 2018 for their help with the organization of the event, and IFIP WG8.1 for the support.

April 2018

Jens Gulden
Rainer Schmidt

EMMSAD 2018

The field of information and software systems development has resulted in a rich heritage of modeling approaches (e.g., business process modeling, enterprise modeling, value modeling, capability modeling, ontology modeling, and so on). This canon of approaches continuous to be enriched with extensions, refinements, and even new languages to deal with new challenges. Even with some attempts toward standardization (e.g., UML for object-oriented software design, ArchiMate for enterprise architecture modeling, and BPMN for business process modeling), new modeling methods are constantly being introduced, especially in order to deal with emerging trends such as compliance and regulations, cloud computing, big data, business analytics, the Internet of Things, cyber-physical systems, etc. These introduce challenges to modeling as well: scalability, privacy, security, and performance, to list a few, and may call for extending existing modeling methods or developing new ones. These ongoing changes significantly impact the way systems are being analyzed and designed in practice.

Evaluation of modeling methods contributes to the knowledge and understanding of their strengths and weaknesses. This knowledge may guide researchers toward the development of the next generation of modeling methods and help practitioners select the modeling methods most appropriate for their needs. A variety of empirical and non-empirical evaluation approaches can be found in the literature: feature comparison, meta-modeling, metrics, paradigmatic analyses, contingency identification, ontological evaluation, surveys, laboratory and field experiments, case studies, and action research. Yet, there is a paucity of such research in the literature.

The objective of the EMMSAD conference series is to provide a forum for researchers and practitioners interested in modeling methods for systems analysis and development (SA&D) to meet and exchange research ideas and results. To this end, the focus is on both insights in modeling for SA&D in general and the fostering of cross-pollination of insights between different specific modeling approaches (such as business process modeling, enterprise modeling, value modeling, capability modeling, etc.). More details can be found at http://www.emmsad.org/.

EMMSAD 2018 accepted six papers that underwent a rigorous review process with four reviewers for each submission. The accepted papers cover a wide spectrum of issues related to modeling:

– “The Power/Generality Trade-Off in Decision and Problem Modeling: Theoretical Background and Multi-Level Modeling as a Resolution”
– “Modeling Organizational Structures in the Realm of Enterprise Modeling: Limitations of the Current Paradigm and Prospects of Multilevel Language Architectures”
– “DevOps Competences and Maturity for Software Producing Organizations”
– “An Agile Modeling Oriented Process for Logical Architecture Design”

– “Exploring the Design Needs for the New Database Era”
– “Evaluation of a Design Method for Graph Database”

We wish to thank the EMMSAD 2018 authors for having shared their work with us, as well as the members of the EMMSAD 2018 Program Committee for their valuable reviews. We also thank the organizers of CAiSE 2018 for their help with the organization of the event, and IFIP WG8.1 for the support.

April 2018

Iris Reinhartz-Berger
Sérgio Guerreiro
Wided Guédria
Palash Bera

BPMDS 2018 Organization

Organizers

Jens Gulden	University of Duisburg-Essen, Germany
Rainer Schmidt	Munich University of Applied Sciences, Germany

Steering Committee

Ilia Bider	Stockholm University and IbisSoft, Sweden
Selmin Nurcan	Université Paris 1 Panthéon - Sorbonne, France
Rainer Schmidt	Munich University of Applied Sciences, Germany
Pnina Soffer	University of Haifa, Israel

Industrial Advisory Board

Ilia Bider	Stockholm University and IbisSoft, Sweden
Pascal Negros	Arch4IE, France
Gil Regev	EPFL and Itecor, Switzerland

Industrial Track Chairs

Rainer Schmidt	Munich University of Applied Sciences, Germany
Jens Gulden	University of Duisburg-Essen, Germany

Program Committee

João Paulo A. Almeida	Federal University of Espirito Santo, Brazil
Judith Barrios Albornoz	University of Los Andes, Colombia
Kahina Bessai	Loria University of Lorraine, France
Ilia Bider	Stockholm University/IbisSoft, Sweden
Karsten Boehm	FH KufsteinTirol - University of Applied Science, Austria
Lars Brehm	Munich University of Applied Science, Germany
Dirk Fahland	Eindhoven University of Technology, The Netherlands
Claude Godart	Loria University of Lorraine, France
Renata Guizzardi	Universidade Federal do Espirito Santo, Brazil
Jens Gulden	University of Duisburg-Essen, Germany
Amin Jalali	Stockholm University, Sweden
Paul Johannesson	Royal Institute of Technology, Sweden
Marite Kirikova	Riga Technical University, Latvia
Agnes Koschmider	Karlsruhe Institute of Technology, Germany
Marcello La Rosa	The University of Melbourne, Australia
Jan Mendling	Vienna University of Economics and Business, Austria

Michael Möhring	Aalen University, Germany
Pascal Negros	Université Paris 1 Panthéon - Sorbonne, France
Jens Nimis	University of Applied Sciences Karlsruhe, Germany
Selmin Nurcan	Université Paris 1 Panthéon - Sorbonne, France
Oscar Pastor Lopez	Universitat Politècnica de València, Spain
Elias Pimenidis	University of the West of England, UK
Gregor Polančič	University of Maribor, Slovenia
Gil Regev	Ecole Polytechnique Fédérale de Lausanne, Switzerland
Manfred Reichert	University of Ulm, Germany
Iris Reinhartz-Berger	University of Haifa, Israel
Stefanie Rinderle-Ma	University of Vienna, Austria
Colette Rolland	Université Paris 1 Panthéon - Sorbonne, France
Michael Rosemann	Queensland University of Technology, Australia
Shazia Sadiq	The University of Queensland, Australia
Rainer Schmidt	Munich University of Applied Sciences, Germany
Stefan Schönig	University of Bayreuth, Germany
Samira Si-Said Cherfi	CEDRIC - Conservatoire National des Arts et Métiers, France
Pnina Soffer	University of Haifa, Israel
Roland Ukor	FirstLinq Ltd., UK
Barbara Weber	University of Innsbruck, Austria
Matthias Weidlich	Humboldt-Universität zu Berlin, Germany
Jelena Zdravkovic	Stockholm University, Sweden
Alfred Zimmermann	Reutlingen University, Germany

Additional Reviewers

Bock, Alexander
Kaes, Georg
de Kinderen, Sybren
Mohring, Tim
Mundbrod, Nicolas
Nolte, Mario
Stach, Michael
Tsoury, Arava
Wang, Wei

EMMSAD 2018 Organization

Co-chairs

Iris Reinhartz-Berger	University of Haifa, Israel
Sérgio Guerreiro	Instituto Superior Técnico/Universidade de Lisboa, Portugal
Wided Guédria	Luxembourg Institute of Science and Technology (LIST), Luxembourg
Palash Bera	Saint Louis University, USA

Advisory Committee

John Krogstie	Norwegian University of Science and Technology (NTNU), Norway
Henderik A. Proper	Luxembourg Institute of Science and Technology (LIST), Luxembourg, and Radboud University Nijmegen, The Netherlands

Program Committee

Palash Bera	Saint Louis University, USA
Tony Clark	Sheffield Hallam University, UK
Dolors Costal	Universitat Politècnica de Catalunya, Spain
Sybren De Kinderen	University of Duisburg-Essen, Germany
Claudio Di Ciccio	Vienna University of Economics and Business, Austria
John Erickson	University of Nebraska-Omaha, USA
Neil Ernst	University of Victoria, Canada
Peter Fettke	German Research Center for Artificial Inteilligence (DFKI) and Saarland University, Germany
Kathrin Figl	Vienna University of Economics and Business (WU), Austria
Mohamad Gharib	University of Florence, Italy
Jeff Gray	University of Alabama, USA
Wided Guedria	LIST
Sérgio Guerreiro	Instituto Superior Técnico, University of Lisbon, Portugal
Stijn Hoppenbrouwers	HAN University of Applied Sciences, The Netherlands
Jennifer Horkoff	Chalmers and the University of Gothenburg, Sweden
Timothy Lethbridge	University of Ottawa, Canada
Florian Matthes	Technical University of Munich, Germany
Raimundas Matulevicius	University of Tartu, Estonia
Haralambos Mouratidis	University of Brighton, UK
Andreas L. Opdahl	University of Bergen, Norway

Sietse Overbeek	Utrecht University, The Netherlands
Hervé Panetto	CRAN, University of Lorraine, CNRS, France
Oscar Pastor Lopez	Universitat Politècnica de València, Spain
Barbara Pernici	Politecnico di Milano, Italy
Anne Persson	University of Skövde, Sweden
Nuno Pombo	University of Beira Interior, Portugal
Jolita Ralyté	University of Geneva, Switzerland
Iris Reinhartz-Berger	University of Haifa, Israel
Alberto Silva	Universidade de Lisboa, Portugal
Sase Singh	Elizabeth City State University, USA
Janis Stirna	Stockholm University, Sweden
Arnon Sturm	Ben-Gurion University, Israel
Dirk van der Linden	University of Bristol, UK
Steven van Kervel	Formetis BV
Carson Woo	The University of British Columbia, Canada
Michael Wufka	Douglas College, Canada
Marielba Zacarias	Universidade do Algarve, Portugal
Anna Zamansky	University of Haifa, Israel
Jelena Zdravkovic	Stockholm University, Sweden

Additional Reviewers

Bhat, Manoj
Borsato, Milton
Detro, Silvana
Li, Qing
Waltl, Bernhard

Contents

Context-Awareness in Business Processes (BPMDS 2018)

Mining Expressive and Executable Resource-Aware Imperative Process Models

Cristina Cabanillas[1(✉)], Stefan Schönig[2], Christian Sturm[2], and Jan Mendling[1]

[1] Vienna University of Economics and Business, Vienna, Austria
{cristina.cabanillas,jan.mendling}@wu.ac.at
[2] University of Bayreuth, Bayreuth, Germany
{stefan.schoenig,christian.sturm}@uni-bayreuth.de

Abstract. Process mining extracts relevant information on executed business processes from historical data stored in event logs. The data typically available include the activities executed, temporal information and the resources in charge of their execution. With such data, the functional, behavioural and organisational perspectives of a process can be discovered. Many existing process mining approaches are capable of generating representations involving the first two perspectives with all types of processes. The extraction of simple and complex resource assignment rules has also been tackled with declarative process models. However, it is noticeable that despite imperative notations like BPMN are mostly used for process modelling nowadays, the existing process mining approaches for enriching such models with resource assignments cannot discover rules like separation of duties and do not produce executable resource-aware process models. In this paper we present an approach for mining resource-aware imperative process models that uses an expressive resource assignment language (RALph) with the de-facto standard notation BPMN. The organisational perspective of the resulting models can be automatically analysed thanks to the formal semantics of RALph. The method has been implemented and tested with a real use case.

Keywords: Organisational mining · Process mining
RALph · Resource assignment · Resource mining

1 Introduction

Process mining extracts relevant information on executed business processes from historical data stored in event logs. The discovered process models can be used for subsequent process improvement or for compliance checking against reference models or regulations [1]. The richer the data in the event logs is, the more facets of the underlying processes can be discovered. Typical data stored

This work was funded by the Austrian Science Fund (FWF) - grant V 569-N31 (PRAIS).

in event logs include the activities executed, temporal information on activity executions and the employees of the organisation in charge of them. With such data, the functional (activities), behavioral (control flow) and organisational (human resources, or for short resources) perspectives of business processes can be discovered [2].

Most of the current support for process mining focuses on the two former perspectives and is capable of generating textual as well as graphical representations of the processes discovered [3,4]. The target of those approaches have been both routine (a.k.a. procedural) processes, usually modelled with imperative notations, such as Business Process Model and Notation (BPMN) [5]; and flexible processes, for which declarative notations, such as Declare [6], are preferred. Work on mining the organisational perspective resulting in declarative process models with expressive resource assignments specifying who is allowed to execute the process activities according to the roles, skills and a number of properties related to the resources and the process, has recently been done [7]. Approaches on resource mining with imperative output models have also been developed [8–10]. However, frequently used rules like separation of duties [11] cannot be discovered, and the graphical resource assignments defined are based on the BPMN swimlanes, which are not provided with semantics [5]. That constrains the expressiveness of the resulting models as well as their use to only documentation purposes and not for automatic execution or analysis. This gap is remarkable as BPMN is the de-facto standard notation for process modelling and is used in most of current Business Process Management Systems (BPMSs).

In this paper we introduce an approach for mining resource-aware imperative process models that uses a graphical notation for modelling resource assignments called RALph [12] together with BPMN. As a result, we obtain an executable specification of a business process that can contain a large variety of resource assignment rules, including those defined in the acknowledged creation patterns [11]. The respective RALphMiner has been implemented using an SQL-based mining technique [13], and it has been tested with a real use case from the university domain.

The remainder of the paper is structured as follows: Sect. 2 presents background information by describing the research problem and related work. Section 3 explains our process mining approach. Section 4 describes our implementation and the application results. Finally, Sect. 5 concludes the paper and gives an outlook of directions for potential future work.

2 Background

In this section, we discuss the background of our work. Section 2.1 describes the research problem that we address and Sect. 2.2 summarises previous work.

2.1 Research Problem

The process participants are the actual responsible for the correct operation of the business processes of an organisation. The specification of *who must do*

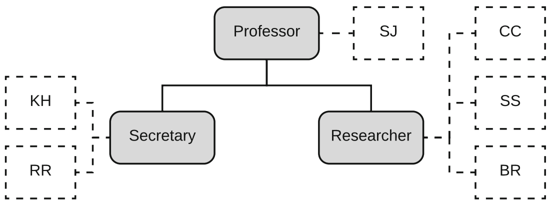


Fig. 1. Hierarchy of positions within a research group

what is usually done based on organisational information. The most common conditions or rules to assign resources to process activities have been collected in a subset of the acknowledged workflow resource patterns, specifically, the creation patterns [11]. They include the following *organisational patterns*: Direct (Dir), Role-based (Rb), Capability-based (Cb), History-based (Hb), Organisational (Org) and Deferred (Def) Distribution; Separation of Duties (SoD); Case Handling (CH); and Retain Familiar (RF), a.k.a. Binding of Duties.

Each organisation may be interested in a certain group of patterns for assigning resources to processes, depending on the organisational information available. Usual data include organisational units, positions, roles and characteristics of the specific people, such as their skills to undertake certain types of tasks. For instance, the research group (organisational unit) depicted in Fig. 1 is structured as a hierarchy of positions. The group is led by a professor (SJ) accountable for the work of two secretaries (KH, RR) and three researchers (BR, SS, CC). One of the most frequent activities related to research is the management of trips for attending conferences, giving invited research talks, and the like. In that process, a researcher first applies for a work trip, which must be approved by their immediate superior. Once approved, the applicant researcher is in charge of booking the accommodation required and of buying the respective transport tickets. Finally, all the documentation is stored by one of the secretaries in order to preserve it for potential future needs, e.g., internal audits.

Process executions are usually stored in event logs, i.e., machine-recorded files that report on the execution of tasks during the enactment of the instances of a given process. In an event log, every process instance corresponds to a sequence (*trace*) of recorded entries, namely, *events*. Each event is defined by a set of *attributes*. These attributes typically involve an explicit reference to the enacted task and to the operating resource [1]. For instance, the following excerpt of a business trip process event log encoded in the XES logging format [14] shows the recorded information of the *start event* of an instance of the activity *Apply for trip* performed by the resource *SS*.

```
<event>
<string key="org:resource" value="SS"/>
<date key="time:timestamp" value="2017-08-06T14:58:00.000+01:00"/>
<string key="concept:name" value="Apply for trip"/>
<string key="lifecycle:transition" value="complete"/>
</event>
```

However, as different activity instances could be executed by different resources, it is necessary to infer the actual resource assignment rules from the

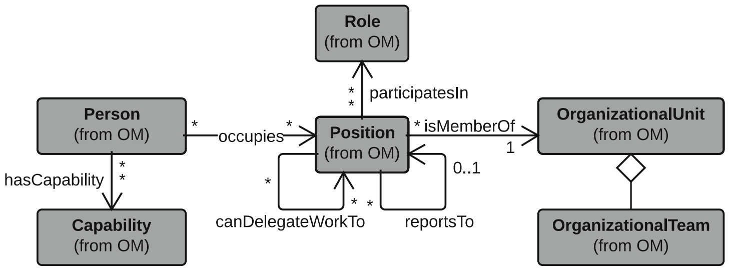


Fig. 2. Organisational metamodel used by RALph (taken from [11])

event log data by applying process mining on the organisational perspective. The organisational information is crucial for that purpose. For instance, in the previous example we should infer that the activity *Apply for trip* is performed by a resource with the position *Researcher*. The output resource assignment rules should be specified together with the functional and behavioural perspectives of the process in the resulting process model.

Therefore, the problem at hand has two inputs, namely, event logs and organisational information. We aim at mining the organisational perspective and defining imperative process models leveraging the fact that BPMN is the de-facto standard notation for process modelling. Furthermore, as BPMN models can be automatised, we want to have executable resource assignment rules, too. Finally, we target simple and complex resource assignment rules on the basis of the organisational patterns. Note that with BPMN, for instance, the last two points are not met since patterns like separation of duties cannot be specified with the BPMN swimlanes and these do not have executable semantics [5]. To address those issues, we can rely on a graphical notation for resource assignment called RALph [12]. The advantages of RALph include: (i) it is independent of any process modelling notation, (ii) it is expressive enough to provide support for all the organisational patterns, and (iii) it has a formal semantics provided by a semantic mapping to Resource Assignment Language (RAL) [15]. RAL is a textual notation whose semantics has been formally defined with description logics, which provides automated analysis power. That means that RALph enables not only the graphical representation of resource assignments, but also their translation to textual assignments as well as their automatic analysis at design time and at run time.

RALph assumes a hierarchical organisation compatible with the organisational metamodel depicted in Fig. 2, similarly to other previous approaches [11]. The notation consists of entities and connectors that enable the visual modelling of resource assignments in process models based on that metamodel. To exemplify the use of RALph, we have modelled some resource assignments of the example scenario in Fig. 3. In Fig. 3a, a *Position* entity is connected to an activity to indicate that a person with the position *Researcher* has to apply for a trip. In Fig. 3b, a RALph *hierarchy connector* is used to specify that the

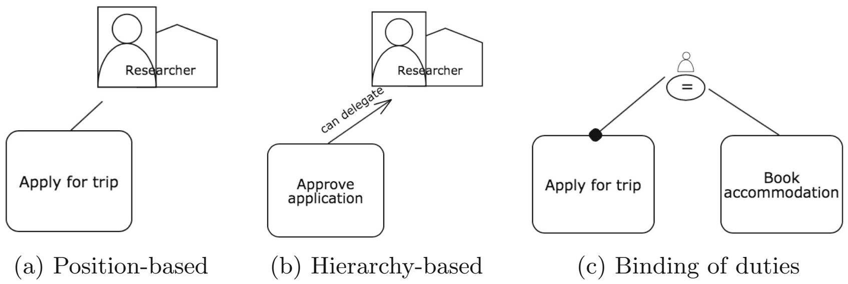


Fig. 3. Examples of RALph assignments with BPMN

Table 1. Approaches for organisational mining: ✓ supported; – not supported; (i) imperative; (d) declarative; (t) textual; (g) graphical; n/a not applicable

Approach	Organisational patterns									Process modelling	Resource assignment	Execution
	Dir	Rb	Def	SoD	CH	RF	Cb	Hb	Org			
[8]	✓	✓	–	–	–	–	✓	–	✓	Petri net (i)	SAR (t)	–
[9, 22]	–	✓	–	–	–	–	–	–	–	n/a	n/a	n/a
[10]	–	✓	–	–	–	–	–	–	–	BPMN (i)	BPMN (g)	–
[7]	✓	✓	–	✓	✓	✓	✓	–	✓	DPIL (d)	DPIL (t)	✓

approval of the application must be done by someone who can delegate work to researchers, i.e., a researcher's superior. Finally, Fig. 3c depicts a binding of duties between two activities, meaning that activity *Book accommodation* has to be executed by the same person who performed activity *Apply for trip*. Note that if we had several assignment rules associated with one activity, the intersection of all of them (AND) would be used to find suitable resources. For more flexibility, RALph provides an *alternative connector* that enables the union of the resource assignment rules (OR). For a more detailed description of the RALph notation we refer to [12].

2.2 Related Work

In the last years, a number of techniques for mining the organisational perspective of a process have been developed [16]. Using input data from process event logs, several methods focus on extracting the organisational model and/or a social network [17] behind a business process, which show the characteristics and relationships among the process participants. There is also increasing interest in analysing resource behaviour and productivity [18] as well as the influence of resources on process performance [19–21].

However, the approaches that are most closely related to our research problem are those which address the discovery of organisational patterns with the aim of enriching a given process model with resource assignments [23] (cf. Table 1).

Among them, the so-called staff assignment mining approach [8] is able to extract several types of assignment rules based on decision tree learning. The identification of separation and binding of duties, among others, are not addressed. The output is an imperative process model (a Petri net, an Event-driven Process Chain (EPC) or a Heuristic net) and textual resource assignments written as Staff Assignment Rules (SAR). The approaches classified as role mining [9,10,22] share with each other the fact that they focus on organisational roles. Some of them address the identification of roles by analysing only the data in the event logs [22]. In this case, resource assignment is not an objective. Others aim at building a Role-Based Access Control (RBAC) model [24] that includes information about roles, permissions and role-based assignments to the process activities [9]. The assignments are part of the RBAC model and hence, they are decoupled from the process model. An explicit link to the process model is present in the approach introduced in [10], which uses the Handover of Roles (HooR) principle to enrich a given control-flow model with roles that cluster the process activities under the assumption that each resource has exactly one role. BPMN and its swimlanes [5] are used to show the outcome of the approach. This and some of the aforementioned methods have been integrated into the ProM tool suite[1]. None of the previous approaches covers the whole set of organisational patterns. The DpilMiner was developed to narrow that gap [7]. It implements a three-step framework that can mine not only most of the organisational patterns but also patterns that consider the control flow and the resources together. The output is a declarative process model with textual resource assignments defined with Declarative Process Intermediate Language (DPIL) [25]. The History-based Distribution pattern is not covered because DPIL does not support the definition of the respective resource assignments. On the other hand, the Deferred Distribution pattern can in no case be addressed by a mining approach as it relates to run time.

From the previous discussion we identify two major challenges related to resource mining with imperative process models: (i) the discovery of a large amount of organisational patterns (i.e., expressiveness), and (ii) an executable and user-friendly specification of the resulting resource-aware process models.

3 RALph Mining Method

In the following, we describe our approach for mining the organisational patterns and generating RALph-aware process models. First, in Sect. 3.1 we outline the fundamental concepts on which the method relies. Afterwards, in Sect. 3.2 we define a set of templates that we need in order to discover RALph assignment rules. In Sect. 3.3 we describe the metrics we use for discovering the rules. In Sect. 3.4 we explain and exemplify the discovery mechanism, based on Structured Query Language (SQL) queries. Finally, in Sect. 3.5 we delve into the order in which certain queries must be run and in how to refine the output RALph-aware process models obtained from the mining.

[1] http://www.promtools.org/doku.php.

3.1 Fundamentals of Multi-perspective Process Mining

The discovery of the behavioural perspective (control flow) of imperative process models uses any kind of mining technique. However, the extension of such models to include or enhance further perspectives is often done through declarative constraints. That is the case of RALph, which declaratively adds resource assignment rules to provide more expressiveness to the organisational perspective. Therefore, we apply declarative mining to address the problem at hand. Declarative mining is based on the definition of *constraint templates*. Templates are patterns that define parameterised classes of properties, and constraints are their concrete instantiations. Constraint templates are used for querying the provided event log to find solutions for the placeholders. A solution, a.k.a. *constraint candidate*, is any combination of concrete values for the placeholders that yields a concrete rule that is satisfied in the event log. This approach has its roots in declarative process modelling notations, based on rules or constraints, especially in Declare [6]. For instance, a *response* constraint indicates that if activity A *occurs*, activity B must eventually *follow*. A template for this constraint parameterises the variable elements of the rule, in this case A and B, so that by replacing these placeholders with specific activities found in traces of an event log, it can be automatically identified which pairs of activities fulfil the constraint. For example, the *response constraint* is fully satisfied in the traces $\mathbf{t}_1 = \langle A, A, B, C\rangle$, $\mathbf{t}_2 = \langle B, B, C, D\rangle$, and $\mathbf{t}_3 = \langle A, B, C, B\rangle$, but not in $\mathbf{t}_4 = \langle A, B, A, C\rangle$ because, in this case, the second occurrence of A is not followed by a B. In $\mathbf{t}_2$, it is actually *vacuously satisfied* [26], i.e., in a trivial way, because A never occurs.

The semantics of the constraints and the templates can be formalised using formal logics, such as Linear Temporal Logic over finite traces (LTL$_f$) [27]. Declare has traditionally focused on the process functional and behavioural perspectives. The operators that have been typically used include, among others, the $\mathbf{F}$ and $\mathbf{G}$ LTL$_f$ future operators, where: $\mathbf{F}\psi_1$ means that ψ_1 holds sometime in the future, and $\mathbf{G}\psi_1$ means that ψ_1 holds forever in the future. The aforementioned *response* constraint is defined with LTL$_f$ as $\mathbf{G}(A \rightarrow \mathbf{F}B)$.

An *activation activity* of a constraint in a trace is an activity whose execution imposes, because of that constraint, some obligations on the execution of other activities (*target activities*) in the same trace. For example, in the *response* constraint A is an activation activity and B is a target activity, because the execution of A forces B to be executed. An activation of a constraint leads to a *fulfilment* or to a *violation*. Consider again $\mathbf{G}(A \rightarrow \mathbf{F}B)$. In the trace $\mathbf{t}_1$, the constraint is activated and fulfiled twice, whereas in trace $\mathbf{t}_3$, it is activated and fulfiled only once. Referring to the formal specification of constraints in LTL$_f$, *activation* ϕ_a is the sub-formula that lies on the left-hand side of the implication operator $\rightarrow$, whereas *target* ϕ_t is the formula that lies on its right-hand side.

The importance of multi-perspective dependencies led to the definition of a multi-perspective version of Declare (MP-Declare) [28], which is of interest to us since we aim at defining templates for constraints that relate to the process organisational perspective. Its semantics build on the notion of *payload* of an event, which is the set of attributes that define it (cf. Sect. 2.1).

$e(activity)$ identifies the occurrence of an event in order to distinguish it from the activity name. At the time of a certain event e, its attributes $x_1, \ldots, x_m$ have certain values. $p^e_{activity} = (val_{x1}, \ldots, val_{xn})$ represents its payload. To denote the projection of the payload $p^e_A = (x_1, \ldots, x_n)$ over attributes $x_1, \ldots, x_m$ with $m \leqslant n$, the shorthand notation $p^e_A[x_1, \ldots, x_m]$ is used. For instance, $p^e_{ApplyForTrip}[Resource] = \text{SS}$ is the projection of the attribute *Resource* in the event description shown in Sect. 2.1. Furthermore, the n-ples of attributes x_i are represented as $\vec{x}$.

Therefore, the templates in MP-Declare extend standard Declare with additional conditions on event attributes. Specifically, given the events $e(A)$ and $e(B)$ with payloads $p^e_A = (x_1, \ldots, x_n)$ and $p^e_B = (y_1, \ldots, y_n)$, the *activation condition* φ_a, the *correlation condition* φ_c, and the *target condition* φ_t are defined. The activation condition is part of the activation ϕ_a, whilst the correlation and target conditions are part of the target ϕ_t, according to their respective time of evaluation. The *activation* condition is a statement that must be valid when the activation occurs. In the case of the *response* template, the activation condition has the form $\varphi_a(x_1, \ldots, x_n)$, meaning that the proposition φ_a over $(x_1, \ldots, x_n)$ must hold true. The *correlation* condition is a statement that must be valid when the target occurs, and it relates the values of the attributes in the payloads of the activation and the target event. It has the form $\varphi_c(x_1, \ldots, x_m, y_1, \ldots, y_m)$ with $m \leqslant n$, where φ_c is a propositional formula on the variables of both the payload of $e(A)$ and the payload of $e(B)$. *Target* conditions exert limitations on the values of the attributes that are registered at the moment wherein the target activity occurs. They have the form $\varphi_t(y_1, \ldots, y_m)$ with $m \leqslant n$, where φ_t is a propositional formula involving variables in the payload of $e(B)$.

3.2 RALph Assignment Templates

Resource assignment modelling languages like RALph are declarative by nature. Therefore, in order to extract RALph-aware process models from event logs, we can rely on existing principles for declarative process mining.

RALph allows for the definition of constraints concerning the assignment of certain resources to activities. Consider a *Direct Assignment* constraint that reflects a constraint on activity a, demanding a, if executed, to be performed by a specific resource res. The respective template comprises placeholders of type *Activity A* as well as *Resource Res*. In Table 2 we provide all RALph constraint templates that should be discovered by our approach according to RALph's expressive power [12]. The table shows the constraint templates, the corresponding semantics in LTL$_f$ and the related payload, i.e., the event attribute that is considered when mining for a certain assignment constraint. In case of the *Direct Assignment* template we have to query the event log for constraints of the shape $\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$, where the target condition $\varphi_t(\vec{x})$ is of the form $p^e_A[Resource] = val$. To discover *Role-based*, *Capability-based*, *Position-based* and *Unit-based* assignment rules, we query for the same semantics as for *Direct Assignments* but we have to consider different payloads that refer to information

Table 2. Semantics of RALph assignment rules. (*) Resp. *canDelegateWorkTo*

Template	LTL$_f$ Semantics	Related payload cond.
Direct Assignment	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^e_A[Resource] = res$
Role-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^e_A[Role] = r$
Pos.-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^e_A[Position] = p$
Cap.-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^e_A[Capability] = c$
Unit-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^e_A[Unit] = u$
Negated Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^e_A[Unit]! = u$
Binding of Duties	$\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$	$p^e_A[Res.] = p^e_B[Res.]$
Separation of Duties	$\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$	$p^e_A[Res.]! = p^e_B[Res.]$
Hierarchy-based Ass.	$\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$	$p^e_A[Res.]$ *reportsTo* $p^e_B[Res.]$(*)

stemming from the organisational model, e.g., $p^e_A[Position]$ to discover position-based assignments as described in the example scenario in Sect. 2.1. A *Binding of Duties* template $\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$ reflects constraints on activity a and b, demanding b, if executed, to be performed by the same resource as activity a. Here, we query the event log for correlation conditions $\varphi_c(\vec{x}, \vec{y})$ on the payloads of the events that correspond to both activities a and b with the specific condition that $p^e_A[Resource] = p^e_B[Resource]$.

For subsequent automated discovery, the analyst will select from the set of predefined constraint templates the ones to be discovered depending, for instance, on the type of organisational information available (e.g., only roles, roles and positions, etcetera).

3.3 Metrics for RALph Mining

Querying with constraint templates provides for every possible combination of concrete values for the placeholders in the templates the number of satisfactions in the event log. Based on the number of satisfactions, two metrics, *Support* and *Confidence*, are calculated, which express the probability of an assignment constraint to hold in the process. *Support* is the number of fulfilments of a constraint divided by the number of occurrences of the condition of a constraint. The *Confidence* metric scales the support by the fraction of traces in the log wherein the activation condition is satisfied. Constraints are considered valid if their *Support* and *Confidence* measures are above a user-defined threshold. For our approach we adopt the most recent definition by Di Ciccio et al. [3]. Here, we only consider the event-based support that is meant to be used for all constraints wherein both activation and target events occur.

As defined in [3], we denote the set of *events* in a *trace* $\mathbf{t}$ of an event log L that fulfil an LTL$_f$ formula ψ as $\models^e_{\mathbf{t}} (\psi)$. The set of all the *events* in *log* L that fulfil ψ are denoted as $\models^e_L (\psi)$. Given a resource assignment constraint Ξ comprising activation ϕ_a and target ϕ_t, we define the event-based support $\mathcal{S}^e_L$ and the event-based confidence $\mathcal{C}^e_L$ as follows:

$$\mathcal{S}^e_L = \frac{\sum_{i=1}^{|L|} \left|\models^e_{\mathbf{t}_i}(\Xi)\right|}{\left|\models^e_L(\phi_a)\right|} \quad (1) \qquad \mathcal{C}^e_L = \frac{\mathcal{S}^e_L \times \left|\models^e_L(\phi_a)\right|}{|L|} \quad (2)$$

3.4 Discovering RALph Assignment Rules with SQL

Our proposed RALph mining method builds on the SQL-based process discovery approach described in [13] because of its versatility towards customisation. With SQL queries it is possible to extract relevant process knowledge from event logs stored in a conventional relational database following the *RelationalXES (RXES)* architecture [29]. The database tables in our case include: (1) one event log table capturing the following event attributes: *EventID* (unique identifier for each recorded event), *TraceID* (unique identifier for the corresponding trace), *ActivityID* (name of the corresponding activity the event refers to), *Time* (date and time the event has occurred) as well as *Resource* (identifier of the performing resource); and (2) tables for the relationships in the organisational metamodel (cf. Fig. 2) storing the organisational information, which results in six tables: HasCapability (*Person*, *Capability*), Occupies (*Person*, *Position*), ReportsTo (*Position*, *Position*) - resp. CanDelegateWorkTo -, ParticipatesIn (*Position*, *Role*) and IsMemberOf (*Position*, *Unit*).

The mining technique discovers all constraints of a certain template under the consideration of two thresholds *minSupp* and *minConf* related to the metrics described in Sect. 3.3 by applying conventional database queries without any parsing or data conversion. As an example, we explain the SQL query to extract *Direct Assignment* constraints, i.e., the fact that a certain activity has been executed by a specific resource.

```
SELECT  'Direct Assignment', A, l1.Resource, [Support], [Confidence]
FROM Log l1, [ActivityCombinations] c
WHERE l1.Activity = c.A
GROUP BY c.A, c.Resource
HAVING [Support] > minSupp AND [Confidence] > minConf
```

In the FROM clause the data source tables are joined together, i.e., the table of the analysed event log where every tuple depicts a single event and, if available, the tables of the *OrganisationalModel*. Furthermore, the clause contains a subquery *ActivityCombinations* that provides a table with the activity combinations that should be checked. Every source table gets an abbreviation assigned to be referable in other clauses, e.g., "l1" for the event log table or "c" for the combination table. The WHERE clause contains the different constraint expressions that have to hold for activities and their events, i.e., the constraint activation condition as well as its fulfilment requirements. After deriving the fulfilments, the tuples are grouped by the set of parameters of the constraint template in the GROUP BY clause. After grouping, the number of tuples corresponding to a certain parameter combination can be extracted using the SQL aggregate function COUNT(*). In addition, a subquery computes the number of occurrences of the condition of the constraint. This way, the *Support* value of each constraint can be derived. The *Confidence* of each parameter combination can be calculated in a similar way. The resulting values of both queries can then be filtered by user-defined thresholds. In the last step, the query output is selected in the SELECT clause, i.e., the parameter combination and its corresponding *Support* $\mathcal{S}^e_L$ and *Confidence* $\mathcal{C}^e_L$ values. The result set contains tuples for each parameter combination that fulfils the constraint under consideration of the given thresholds. The *Support* value is computed with the subquery below.

```
COUNT(*) /  (SELECT COUNT(*) FROM Log WHERE Activity = A)
```

We next explain the query to extract *Position-based Assignment* constraints. The FROM, WHERE and GROUP BY clauses of the query are as follows:

```
SELECT  'Position-based Assignment', A, l1.Unit, [Support], [Confidence]
FROM Log l1, Position p, [ActivityCombinations] c
WHERE l1.Activity = c.A AND a.Resource = u.Resource
GROUP BY c.A, p.Position
HAVING [Support] > minSupp AND [Confidence] > minConf
```

In addition to the event log and the activity combinations we also join the table with the resource-positions assignments according to the organisational model in the FROM clause. The query sums up all occurrences of events with respective resources and groups the occurrences w.r.t. the corresponding position given in the table *Occupies*.

This approach is followed to define SQL queries for all the types of resource assignments that we aim at discovering, in our case, those in Table 2. All the queries can be found in [30].

3.5 Alternative Connectors and Pruning

If with certain *minSupp* and *minConf* thresholds we do not extract any resource assignment rule for a process activity, it could be the case that several resource assignment rules are associated to it with lower frequencies. Consider, for instance, that for an activity *Apply for trip* we could not extract a valid *Position-based Assignment* rule since for no rule candidate $\mathcal{S}^e_L > minSupp$ with, e.g., $minSupp = 0.95$ holds. In this case, however, it could be possible to extract a *Position-based Assignment* rule for *Researcher* with $\mathcal{S}^e_L = 0.5$ and a *Capability-based Assignment* rule for *Can speak English* with $\mathcal{S}^e_L = 0.5$, respectively. This union is modelled with the RALph alternative connector to express that one of the two conditions suffices to find suitable resources. Therefore, alternative connectors are examined at the end of the mining procedure using lower support thresholds and combining the different extracted assignment rules.

The mining method extracts *all* the assignment rules related to each activity. However, when several rules are extracted for one single activity (AND), not all of them might be strictly necessary to understand the process. Specifically, some rules may be implied by *stronger* rules because they are less restrictive and do not provide added value to the current resource assignment expression of an activity. Those rules complicate the understandability of the discovered

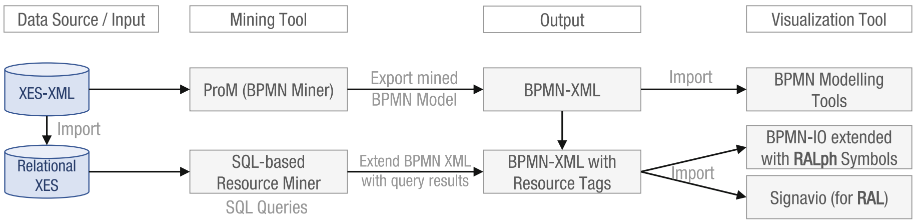


Fig. 4. Implementation architecture and mining procedure

models and hence, they are unnecessary. The work in [7] identifies two pruning approaches to eliminate unnecessary resource assignment rules: pruning based on organisational rule hierarchies (e.g., *position-based assignment* dominates *direct assignment*) and pruning based on transitive reduction (e.g., for binding of duties rules). The requirement for all pruning operations is that they do not change the meaning of the generated model. These post-processing methods can be applied to the approach at hand in a similar way in order to avoid overloading the output RALph-aware process models with unnecessary assignments that would, on the other hand, worsen their readability.

4 Implementation and Evaluation

The RALph mining approach has been implemented as a web-based process mining tool. The implemented architecture and used toolset are illustrated in Fig. 4. We aim at discovering RALph-aware process models and hence, two main elements must be discovered, namely, the definition of the process itself (i.e., the functional and behavioural perspectives) and the resource assignment rules for the process activities (i.e., the organisational perspective). As mentioned in Sect. 1, there is a number of approaches for discovering a business process. Implementations for many of them are available as plug-ins in the ProM framework. We use the *BPMN Miner* tool with a XES event log to extract a resource-unaware BPMN model. Afterwards, the resulting BPMN model is exported as an XML file according to the *BPMN-XML* specification [5]. We use the SQL mining approach described in Sect. 3.4 for extracting RALph assignment rules. Since this approach builds on the relational RXES event log representation, we first have to import the XES event log to relational database tables in RXES format as well as make the organisational information available as tables as explained in Sect. 3.4. We can then run the set of SQL queries required to extract RALph resource assignment rules. The resulting assignment rules are attached in the previous BPMN-XML file to the respective activity as specific resource tags. Here, we match activities from the given BPMN model and the extracted assignment rules based on activity identifiers given in the event log. The RALph-aware BPMN model is then visualised in the graphical BPMN diagram editor *bpmn.io*[2], which has been extended with the RALph symbols. For automatically

[2] BPMN Viewer and Editor, https://bpmn.io.

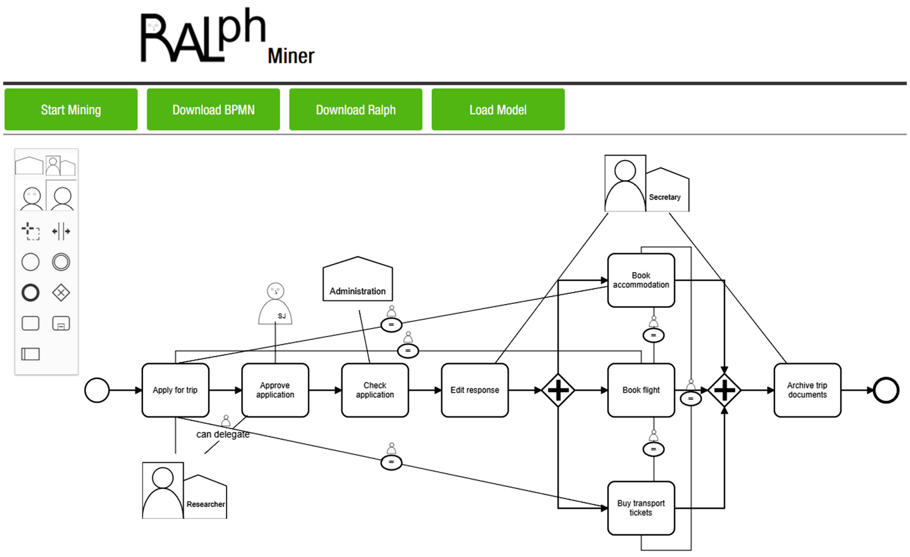


Fig. 5. User interface of the RALph miner with extracted assignment rules

arranging and layouting the RALph assignment symbols in the process diagram, we used a Java Script based implementation of the Sugiyama graph layout algorithm [31]. Additionally, the underyling formal RAL expressions can be imported and edited in BPMN editors like *Signavio*[3]. A plug-in is available to automatically analyse such RAL assignments so that the RAL-aware process model can be automatically executed [32].

As a proof of concept, we applied the described toolset to an event log of a university business trip management system. The log contains 2104 events of 8 different activities related to the application and the approval of university business trips as well as the management of accommodations and transfers, e.g., booking accommodations and transport tickets. The system has been used for 6 months by 11 employees of a research institute. The organisational model of the institute comprises 2 organisational units: Administration, with 2 employees; and Research Group, divided into 3 positions that include 6 researchers, 1 professor and 2 secretaries. Since the underlying mining technique is based on SQL queries, the performance of our approach is directly related to the corresponding mining approach introduced in [13]. On the given event log, we were able to execute all RALph resource assignment queries (cf. Table 2) in less than one second. The resulting BPMN model with the extracted RALph assignment rules is shown in Fig. 5. The screenshot also shows the extended *bpmn.io* modelling toolbox on the left-hand side. Note that the model has not been pruned as described in Sect. 3.5 since the implementation of that post-processing feature is still pending work. Therefore, the model contains some assignment rules that are irrelevant,

[3] https://www.signavio.com.

e.g., the direct assignment of entity *SJ* to *Approve Application* or the binding of duties rule between *Book accommodation* and *Buy transport tickets*.

5 Conclusions and Future Work

In this paper we have addressed a gap in process mining related to the extraction of expressive resource assignment conditions from event logs and the generation of executable resource-aware imperative process models as output models. We have legeraged existing methods and notations, in particular: an SQL-based process mining approach has been extended and the RALph notation has been used to enrich the resulting BPMN process models.

The limitations of the work presented here include: (i) the syntactic translation from LTL$_f$ to SQL has been omitted in this paper due to space limitations but it has been performed as part of a follow-up contribution; and (ii) despite the organisational metamodel used in RALph has been extensively used in other approaches, in companies with a different structure some aspects of the organisation might not be captured and considered.

The implementation of the pruning step after the discovery of the RALph-aware process models is the next task to be performed. Afterwards, an extended evaluation of the quality of the approach will be conducted. Besides, we aim to investigate whether RALph could be used to add resource assignments to declarative process modelling notations for increasing expressiveness. Scalability in resource-aware process models is also an issue to be addressed in this context.

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19345-3
2. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
3. Di Ciccio, C., Mecella, M.: On the discovery of declarative control flows for artful processes. ACM Trans. Manag. Inf. Syst. **5**(4), 24:1–24:37 (2015)
4. Rovani, M., Maggi, F.M., de Leoni, M., van der Aalst, W.M.: Declarative process mining in healthcare. Expert Syst. Appl. **42**(23), 9236–9251 (2015)
5. OMG: BPMN 2.0, recommendation, OMG (2011)
6. van der Aalst, W., Pesic, M., Schonenberg, H.: Declarative workflows: balancing between flexibility and support. Comput. Sci. R&D **23**(2), 99–113 (2009)
7. Schönig, S., Cabanillas, C., Jablonski, S., Mendling, J.: A framework for efficiently mining the organisational perspective of business processes. Decis. Support Syst. **89**, 87–97 (2016)
8. Rinderle-Ma, S., van der Aalst, W.M.: Life-cycle support for staff assignment rules in process-aware information systems, Technical report, TU/e (2007)
9. Baumgrass, A.: Deriving current state RBAC models from event logs. In: International Conference on Availability, Reliability and Security, pp. 667–672 (2011)

10. Burattin, A., Sperduti, A., Veluscek, M.: Business models enhancement through discovery of roles. In: IEEE CIDM, pp. 103–110 (2013)
11. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Workflow resource patterns: identification, representation and tool support. In: Pastor, O., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 216–232. Springer, Heidelberg (2005). https://doi.org/10.1007/11431855_16
12. Cabanillas, C., Knuplesch, D., Resinas, M., Reichert, M., Mendling, J., Ruiz-Cortés, A.: RALph: a graphical notation for resource assignments in business processes. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 53–68. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_4
13. Schönig, S., Rogge-Solti, A., Cabanillas, C., Jablonski, S., Mendling, J.: Efficient and customisable declarative process mining with SQL. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 290–305. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39696-5_18
14. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 60–75. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-17722-4_5
15. Cabanillas, C., Resinas, M., del Río-Ortega, A., Ruiz-Cortés, A.: Specification and automated design-time analysis of the business process human resource perspective. Inf. Syst. **52**, 55–82 (2015)
16. Bose, R.P.J.C., Maggi, F.M., van der Aalst, W.M.P.: Enhancing declare maps based on event correlations. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 97–112. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_9
17. Song, M., van der Aalst, W.M.: Towards comprehensive support for organizational mining. Decis. Support Syst. **46**(1), 300–317 (2008)
18. Pika, A., Leyer, M., Wynn, M.T., Fidge, C.J., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Mining resource profiles from event logs. ACM Trans. Manag. Inf. Syst. **8**(1), 1:1–1:30 (2017)
19. Nakatumba, J., van der Aalst, W.M.P.: Analyzing resource behavior using process mining. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 69–80. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12186-9_8
20. Hompes, B.F.A., Maaradji, A., La Rosa, M., Dumas, M., Buijs, J.C.A.M., van der Aalst, W.M.P.: Discovering causal factors explaining business process performance variation. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 177–192. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_12
21. Wynn, M.T., Poppe, E., Xu, J., ter Hofstede, A.H.M., Brown, R., Pini, A., van der Aalst, W.M.P.: ProcessProfiler3D: a visualisation framework for log-based process performance comparison. Decis. Support Syst. **100**, 93–108 (2017)
22. Jin, T., Wang, J., Wen, L.: Organizational modeling from event logs. In: International Conference on Grid and Cooperative Computing (GCC), pp. 670–675 (2007)
23. Zhao, W., Zhao, X.: Process mining from the organizational perspective. Adv. Intell. Syst. Comput. **277**, 701–708 (2014)
24. American National Standards Institute, Inc.: Role-Based Access Control. ANSI INCITS 359–2004, February 2004. http://csrc.nist.gov/rbac

25. Zeising, M., Schönig, S., Jablonski, S.: Towards a common platform for the support of routine and agile business processes. In: IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, pp. 94–103 (2014)
26. Kupferman, O., Vardi, M.Y.: Vacuity detection in temporal model checking. Int. J. Soft. Tools Technol. Transfer **4**(2), 224–233 (2003)
27. Montali, M., Pesic, M., van der Aalst, W.M.P., Chesani, F., Mello, P., Storari, S.: Declarative specification and verification of service choreographies. TWEB **4**(1), 3:1–3:62 (2010)
28. Burattin, A., Maggi, F.M., Sperduti, A.: Conformance checking based on multi-perspective declarative process models. Expert Syst. Appl. **65**, 194–211 (2016)
29. van Dongen, B.F., Shabani, S.: Relational XES: data management for process mining. In: CAiSE Forum 2015, pp. 169–176 (2015)
30. Schönig, S.: SQL Queries for Declarative Process Mining on Event Logs of Relational Databases, CoRR, vol. abs/1512.00196 (2015)
31. Eiglsperger, M., Siebenhaller, M., Kaufmann, M.: An efficient implementation of Sugiyama's algorithm for layered graph drawing. In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 155–166. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31843-9_17
32. Cabanillas, C., del Río-Ortega, A., Resinas, M., Cortés, A.R.: CRISTAL: collection of resource-centric supporting tools and languages. In: BPM (Demos), pp. 51–56. CEUR-WS.org (2012)

An Integrated Architecture for IoT-Aware Business Process Execution

Stefan Schönig[1](✉), Lars Ackermann[1], Stefan Jablonski[1], and Andreas Ermer[2]

[1] Institute for Computer Science, University of Bayreuth, Bayreuth, Germany
{stefan.schonig,lars.ackermann,stefan.jablonski}@uni-bayreuth.de
[2] Maxsyma GmbH and Co. KG, Floß, Germany
aermer@maxsyma.de

Abstract. Business processes are frequently executed within application systems that involve humans, computer systems as well as objects of the Internet of Things (IoT). While several works are emerging on combining BPM and the IoT, the exploitation of IoT technology for system supported process execution is still constrained by the absence of a common system architecture that manages the communication between both worlds. In this paper, we introduce an integrated approach for IoT-aware business process execution that exploits IoT for BPM by providing IoT data in a process-aware way, providing an IoT data provenance framework, considering IoT data for interaction in a pre-defined process model, and providing wearable user interfaces with context specific IoT data provision. The approach has been implemented and evaluated extensively in production industry. The results show that the application of IoT enhanced BPM leads to less machine stops.

Keywords: Process execution · Internet of Things
Wearable interface

1 Introduction

Business process management (BPM) is considered as powerful technology to operate, control, design, document, and improve cooperative processes [1]. Processes are executed within application systems that are part of the real world involving humans, cooperative computer systems as well as physical objects [2]. Internet of Things (IoT), denoting the inter-networking of all kinds of physical devices, has become very popular these days. It enables continuous monitoring of phenomena based on sensing devices, e.g., wearables, machine sensors, etc. Therefore, IoT contributes to the recent trend known as big data. Process execution, monitoring and analytics based on IoT data can enable a more comprehensive view of processes. Embedding intelligence by way of real-time data gathering from devices and sensors and consuming them through BPM technology helps businesses to achieve cost savings and efficiency.

Let us consider a production process where raw material is processed by different machines under the supervision of human operators. In case of product quality issues, manual human involvements are necessary. Additionally, operators must be aware of current sensor data to decide on tasks to be executed next. Such a scenario might be better manageable when closely linking digital production and machine data with human operators as enabled by the integration of IoT and BPM. Here, the necessity of human activities can be triggered by a BPM engine through the reference of appropriate IoT sensor data in the underlying process model. This way, human operators can be notified seamlessly without loss of time on wearable devices while leveraging current context specific information. As a consequence, the integration of IoT and BPM technology could lead to efficiency gains by reducing reaction time and enhance the quality of task execution.

In literature, several works are emerging on combining IoT and BPM, e.g., monitoring running processes to align them with the state of the things [3,4]. Still, there are many open challenges to be tackled [5]. In particular, the exploitation of IoT technology for system supported process execution is constrained by the absence of a common architecture that manages and standardizes the communication between both worlds. In this paper, we close this gap by proposing an integrated approach for IoT-aware business process execution that exploits IoT for BPM by *(i)* providing IoT data in a process-aware way, *(ii)* providing an IoT data provenance framework, *(iii)* considering IoT data for interaction in a pre-defined process model, and *(iv)* providing wearable user interfaces with context specific IoT data provision. The approach has been implemented and evaluated extensively in production industry. The results show that the application of IoT enhanced BPM leads to less machine stops because operators need less time to recognize work to be done.

This paper is structured as follows: Sect. 2 describes research background and fundamentals. Section 3 introduces the approach for a IoT-aware process execution. In Sect. 4 we describe the implementation of our approach based on well-known communication protocols. In Sect. 5 we evaluate our approach with an extensive application in production industry. Section 6 gives an overview of related work and Sect. 7 finally concludes the paper.

2 Research Background and Fundamentals

The IoT is the inter-networking of physical objects like *(i)* electronic hardware, e.g., sensors and actuators or *(ii)* humans using wearable digital devices like glasses and smartwatches. Such connected things collect and exchange data between each other. IoT allows things to be controlled remotely across existing network infrastructures, including the Internet [6]. A business process is a collection of related events, activities, and decisions that involve a number of (human) resources. To support processes at an operational level, a BPM system (BPMS) can be used. During process execution, a variety of information is required to make meaningful decisions. With the emergence of IoT, data is generated from physical devices sensing their (business/manufacturing) environment that reflects certain aspects of operative processes. A BPMS deals, a.o., with the enactment of process models that define the interplay between environmental circumstances (depicted as data values), characteristics of participants (depicted as resource assignments) and corresponding activities to be executed. We consider processes as explicit process representations (pre-defined models), which later are enacted.

Accordingly, sensing and perception the process environment via sensors constitutes the fundamental task of the IoT. Sensor data then must be aggregated, interpreted and made available to the BPMS in order to trigger business process activities or human tasks, respectively. These tasks must then be send in real time to responsible individuals that receive tasks on mobile user interfaces.

2.1 Research Problem

Recently, several researchers raised specific research challenges that need to be tackled in order to align IoT and BPM technology [5]. In this section, we explain a specific subset of those challenges that is tackled by the integration of IoT objects and a BPMS as it is described in the work at hand.

First of all, IoT sensors need to be placed in a process-aware way in order to be able to collect and record all process relevant IoT data. Therefore, sensors need to be carefully placed at machinery and humans and be digitally accessible. The acquired process relevant IoT information needs to be up-to-date and current. It needs to be clear where the data stems from and where it has been used (cf. *data provenance*), as well as the quality of the data at-hand needs to be ensured. It becomes necessary to find a way to annotate the IoT data's origin and to use this (meta-)information in business process models.

In many processes some activities require the interplay between human operators and software/hardware modules. Furthermore, there is an increasing use of mobile devices fostering the delivery of work items to the right users. Here, an appropriate mapping from activities to user interfaces is needed allowing process participants to perform their work from arbitrary places in the working environment. Participants might suffer from issues which hinder optimal working conditions. Here, the IoT can support the execution of tasks in a process through context-specific knowledge provisioning that is relevant for the users particular context. Sensor data can be leveraged to determine the actual context and to identify information needs.

2.2 Motivating Example

Additionally, we want to motivate the necessity of our approach by example of a real life process from production industry that is visualized in Fig. 1. The example stems from the corrugation area where paper is glued to corrugated paper that depicts the basis for cardboard boxes. The given example is a subprocess that is executed every time paper source rolls run empty, i.e., where new paper rolls need to be spliced with the paper from the low running roll. In order to effectively execute this process, several real time interactions with IoT devices, i.e., sensors and operator equipment, is necessary: the BPMS must be aware of sensor data which indicates that a splice will happen soon, triggering the splice

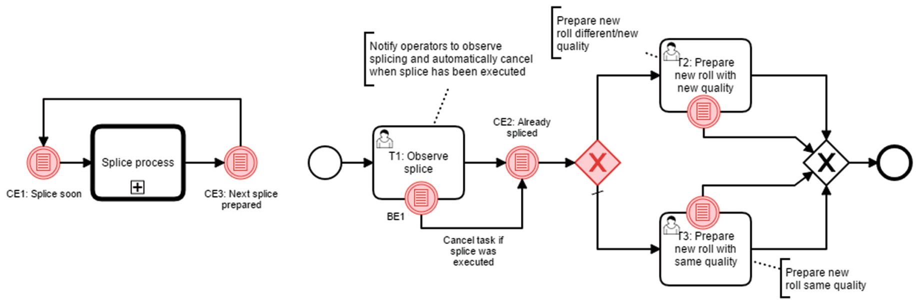

Splice process
CE1: Splice soon
CE3: Next splice prepared
Notify operators to observe splicing and automatically cancel when splice has been executed
T1: Observe splice
BE1
Cancel task if splice was executed
CE2: Already spliced
Prepare new roll different/new quality
T2: Prepare new roll with new quality
T3: Prepare new roll with same quality
Prepare new roll same quality

Fig. 1. Exemplary production process: (a) repeating subprocess; (b) splice process

subprocess. Operators located somewhere along the up to 300 m long machinery need to observe the splice process to avoid issues. Therefore, they need to be notified in real time to walk to the splicer. This requires wearable interfaces communicating with the BPMS over the IoT. Depending on a sensor value indicating the next roll quality, the BPMS has to execute different paths. In case the environment changes, operators tasks need to be reordered based on priorities or cancelled by the BPMS. In addition to current tasks to be executed, operators require context specific information at hand, e.g., the location of the splicer and the quality of the next paper roll. Furthermore, operators continuously need to observe viscosity and temperature of the glue to ensure a successful splice process. In the following sections we show that the integration of IoT devices and a BPMS serves as a generalisable solution to the problems above.

3 Integrated Architecture for an IoT-Aware BPMS

We propose a four-steps procedure to provide the necessary information. The first step is connecting IoT objects and their values traceable to a BPMS. We call a single type of value of a certain (sensor) object *variable*. The second step is extending a BPMN process model with data variables participating in the process stemming from physical objects such as machine status or actor positions. The resulting process models must be applicable by default contemporary BPMN execution engines. This way, organizations can reuse existing process models, without having to learn new languages and remodel processes from scratch. The third step is to establish a real time notification interface of triggered activities to process participants by means of mobile devices. In the fourth step, context relevant information stemming from connected objects must be selected and provisioned to users. Our approach poses the following four main requirements.

R1. *A BPMS must be aware of current values of IoT objects.* Variable attributes, e.g., address and identifiers, must be configurable and traceable, i.e., it must be clear where the data stems from.

R2. *Each defined variable must be referenceable in the executed process model.* Based on current values of certain variable, tasks are triggered or cancelled and decisions are made.

R3. *Responsible users must be notified on mobile devices in real time.* Process participants must be seamlessly notified when human interaction is required, independent of where the user is located.

R4. *Context-specific knowledge must be provisioned to users.* Alongside with activity notification, context-specific and process relevant information must be provisioned to users.

3.1 Connecting Current Data of IoT Objects to BPMS

First, we formally define IoT object data that is regularly, i.e., in a certain interval int acquired from digitally accessible devices and stored in a IoT middleware database. Data recorded from IoT objects is an ordered sequence σ of events e_j, i.e., $\sigma = \langle e_1, ..., e_{|\sigma|} \rangle$. In general, an event e is defined as tupel of attributes-value pairs, i.e., $e = \Big((attr_1^e, val_1^e), ..., (attr_{|e|}^e, val_{|e|}^e) \Big)$. The set of all attributes $ATTR^e$ and the corresponding values VAL^e of an event e are therefore defined as $ATTR^e = \Big\{ attr_1^e, ..., attr_{|e|}^e \Big\}, VAL^e = \Big\{ val_1^e, ..., val_{|e|}^e \Big\}$. Each $attr_i^e$ has an unique variable identifier var and a timestamp $time$, i.e., $\forall_{e \in \sigma}(var^e, time^e) \subseteq ATTR^e$. Each attributes-value pair $(attr_i^e, val_i^e)$ is dedicated to exactly one IoT variable. With $l^e(var_i^e) = val_i^e$ we denote the value val_i^e of a certain variable var_i^e in event e. A total order of events is implemented as follows: $\forall_{1 \leq j < k \leq |\sigma|}(time^{e_j}) < (time^{e_k})$. Therefore, the current value of a certain variable var_i is given as $l^e(var_i^{e_{|\sigma|}}) = val_i^{e_{|\sigma|}}$. All other attributes are optional. Based on these definitions, we ensure that IoT data variables are configurable and traceable (cf. R1). In the next step, current data of each connected IoT object must be sent to the BPMS (cf. R1). Therefore, we need to map IoT data to enacted process models of the BPMS. Consider a set of process models P where each p contains a set of data variables D_p. Each variable $d \in D_p$ has an unique identifier var_d. The underlying assumption is that each participating IoT variable is referenced by the same unique identifier in the corresponding process model. If we want to establish a connection between the data of an IoT object and a process variable then both identifiers have to be the same, i.e., $var_i = var_d^p$. Having established such a semantic mapping, only current data of each connected object is sent to the BPMS, i.e., a sending procedure sp is initiated for each variable $var_i^{e_{|\sigma|}}$ recorded in $e_{|\sigma|}$ sending the latest acquired values to the BPMS, i.e., $\forall_{1 \leq i \leq |e|} val_i^{e_{|\sigma|}}$.

Example: Let us assume, we acquire data from a sensor indicating the restmeters of paper on a specific roll RM_1, the quality of the paper roll to be used next QU and the current glue temperature GT from a temperature sensor in an interval $int = 10\,\text{s}$. The acquired IoT data is then exemplarily given as shown in Table 1. Consider a process model p where two data variables are referenced

Table 1. Example acquired IoT object data

Event	$attr_i^e(var, ..., time)$	val_i^e	Data type
1	(RM_1, ..., 2018-01-09 10:15:32)	300	Historic
1	(QU, ..., 2018-01-09 10:15:32)	211 C	Historic
1	(GT, ..., 2018-01-09 10:15:32)	120	Historic
2	(RM_1, ..., 2018-01-09 10:15:42)	133	Current
2	(QU, ..., 2018-01-09 10:15:42)	211 C	Current
2	(GT, ..., 2018-01-09 10:15:42)	115	Current

and evaluated. These variables are defined as $var^p_{d_1} = RM_1$ and $var^p_{d_2} = QU$, respectively, in the model. The procedure sp is initiated every 10 s. Therefore, the last execution of sp sends the values 133 and $211C$ to the BPMS where these values are dedicated to the corresponding process variables $\{d_1, d_2\} \in D_p$. Note, that data referring to GT is not sent to the BPMS.

3.2 Enrichment of Process Models with IoT Variables

As described in Sect. 3.1, we consider a set of process models P where each p contains a set of process variables D_p that semantically correspond to a variable of an IoT object. Based on the established real time connection described before, these IoT aware process variables, identified by $var^p_{d_i}$, can be referenced in enacted process models in various ways. We assume that a given process model p is defined as BPMN conform process diagram, one of the most used formalisms for process modeling, representing the IoT aware process to be executed. A BPMN process diagram specifies which activities are executed in a process as well as their control flow relationships. To be able to infer when activities start or end based on the state of the variables, the diagram must capture this information (cf. R2). This step requires the process designer to enrich the given BPMN diagram by including information on how and where the connected IoT variables influence the process. We consider several possible interaction possibilities:

a. IoT based trigger events. Sometimes we only want a process to start or to continue if a certain condition is true. In BPMN conditional events define an event which is triggered if a given condition is evaluated to true. It can be used as an *(i) Intermediate Conditional Event* ; as a *(ii) Boundary Event* or as a *(iii) Conditional Start Event.* In case of *(i)* current values of used IoT variables $var^p_{d_i}$ trigger an *Intermediate Catch Event* and the execution continues to the next activity. In case of *(ii)* the BPMS checks if the process environments changed based on the current values of the connected IoT variables. If the given condition is satisfied, then the corresponding activity will be interrupted.

Example: Consider the example process in Fig. 1 where the control flow reaches the conditional event *CE1* and the condition is, for example, $RM_1 < 200$. Given a current IoT sensor value of $val^{e|\sigma|}_{RM_1} = 133$ the condition will be satisfied and the BPMS triggers the subprocess *Splice process.* Similarly, if the splice has

```
**Input**: $T$: Set of available tasks $T = \{t_1, \ldots, t_{|T|}\}$
**Input**: $U$: Set of registered users $U = \{u_1, \ldots, u_{|U|}\}$

foreach $u \in U$ do
    $deviceId$: identifier of mobile device dedicated to $u$
    $tasksToSend \leftarrow \emptyset$
    foreach $t \in T$ do
        if $u$ among candidate users of $t$ then
            $tasksToSend \leftarrow tasksToSend \cup \{t\}$
    if $tasksToSend > 0$ then
        $sendTasks(deviceId)$
```

Algorithm 1. Distributing assigned human tasks to wearable devices

been executed successfully, i.e., $val_{RM_1}^{e_{|\sigma|}} > 200$, then *Observe splice* is aborted by means of the corresponding boundary event *BE1*.

b. IoT based decisions. In BPMN an data-based gateway, is used to model a decision in the process. Similar to conditional events, current IoT variable values can be used to decide which sequence flow is selected for continuing the process.

Example: Depending on the next paper quality to be used in the production order the BPMS has to decide how to continue the process, i.e., based on the current value of $val_{QU}^{e_{|\sigma|}}$ either task *T2* or task *T3* will be triggered.

c. IoT based loops. In order to model repeated behavioural patterns, IoT variables can be used to define event-driven loops. This way, end-to-end processes can be broken up into comprehensible *micro-processes.*

Example: In order to support the recurring splice process in Fig. 1(a), the subprocess is surrounded by the conditional events *CE1* and *CE3*. While the positive evaluation of the given conditions in *CE1*, i.e., splice happening soon, triggers the subprocess to be executed, *CE3* checks if all preparations for the next splice have been executed. In case the corresponding IoT stemming data values fulfil the given event condition, the process continues *CE1*.

3.3 Establishing the Real Time Mobile User Interface

It is important that process participants are seamlessly notified when human interaction is required, independent of where the user is located. This requires a real time notification on mobile devices of responsible users (cf. R3). Therefore, it is necessary to define a mapping of users to corresponding mobile devices that serve as wearable, user-specific process cockpits and task lists, respectively. This is achieved by specifying a dedicated mobile device identifier for each defined user in the BPMS. During process execution, the currently available tasks for a specific process participant are then directly sent to the specified mobile device. The algorithm to distribute the currently available human tasks to defined mobile devices is given in Algorithm 1. The actual operator to device mapping and the task distribution is described in detail in Sect. 4.

Example: The splice process in Fig. 1(b) requires operators to manually observe the splice at the specific machinery (*T1*). Therefore, in case the condition in

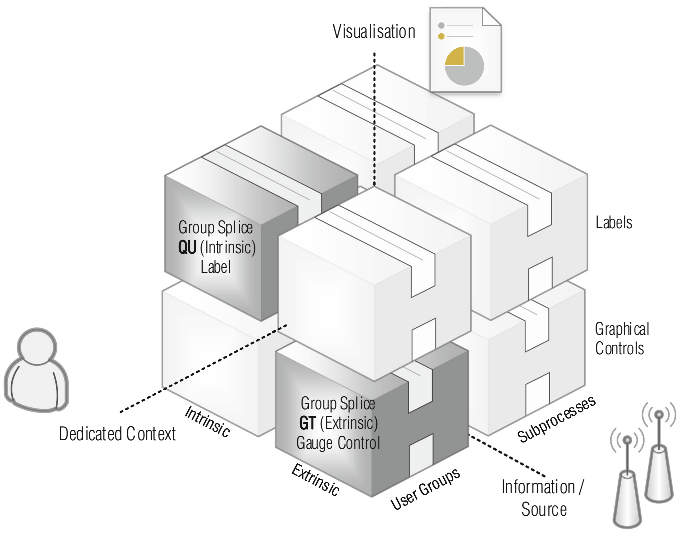


Fig. 2. Three dimensions of context-specific IoT information provision

CE1 evaluates true, the responsive operator needs to be notified in real time to be able to walk to the splice site in time. Hence, as soon as task *T1* becomes available, the list of currently available tasks of assigned operators, implemented as a smartwatch application is updated.

3.4 Context-Specific Information Provision

Alongside with activities, context-specific and process information must be provisioned to operators to improve the quality of task execution (cf. R4). In order to ensure that the provisioned information is of value for operators, the following three dimensions need to be considered when defining data that should be delivered to certain process participants (cf. Fig. 2). Based on these dimensions, Algorithm 2 distributes IoT information in a context aware way to corresponding users.

a. Dedicated Context - Which entities allow for a separate context definition? There are different entities where different contexts can be defined for. IoT variable information can be dedicated to subprocesses and delivered alongside with tasks of this subprocess to respective users. Furthermore, IoT variable data can be dedicated to specific user groups or roles. Within a location-aware BPM scenario, the information context can also be defined based on locations.

b. Information/Source - Which IoT information should be provisioned? IoT data can be classified according to the factors of the context framework defined in [7], i.e., *intrinsic* or *extrinsic* context information. Intrinsic IoT variable information reflects data used in the process model and is therefore directly related to the

```
Input: $C$: Set of defined contexts $C = \{c_1, \ldots, c_{|C|}\}$
Input: $U$: Set of registered users $U = \{u_1, \ldots, u_{|U|}\}$
Input: $V_I$: Set of intrinsic variables $V_I = \left\{v_{i1}, \ldots, v_{|V_I|}\right\}$
Input: $V_E$: Set of extrinsic variables $V_E = \left\{v_{e1}, \ldots, v_{|V_E|}\right\}$
foreach $c \in C$ do
    $relVars_c$: set of relevant attributes-value pairs for context $c$
    $relVars_c \leftarrow \emptyset$
    foreach $v_i \in V_I$ do
        if $v_i$ among related variables of $c$ then
            $val_{v_i} \leftarrow$ receive current value from workflow engine
            $relVars_c \leftarrow relVars_c \cup \{(v_i, val_{v_i})\}$
    foreach $v_e \in V_E$ do
        if $v_e$ among related variables of $c$ then
            $val_{v_e} \leftarrow$ receive current value from IoT middleware
            $relVars_c \leftarrow relVars_c \cup \{(v_e, val_{v_e})\}$
    foreach $u \in U$ do
        $deviceId$: identifier of mobile device dedicated to $u$
        if $u$ among related users of $c$ then
            $sendInformation(relVars_c, deviceId)$
```

Algorithm 2. Provisioning context-specific IoT information

currently executed process instance. Extrinsic IoT data is information that is not necessarily related to a running processes but might influence process execution as well, e.g., production hall temperature. Furthermore, the granularity of provisioned information must be adjustable w.r.t. different processes. In some cases it might be helpful to see more information, in some cases it might be better to see less information to prevent users from information overload.

c. Visualisation - How is a certain context-specific IoT variable presented or visualised? Context-specific IoT information must be visualised in a proper way. Depending on the class of information (intrinsic, extrinsic) or the data type (string, number) different positions on the interface and representation styles are appropriate. Intrinsic, instance specific information that is relevant for the execution of a certain activity can be represented as information below the activity name. Extrinsic, environmental data that might be of importance to process execution, can be represented as separate controls on additional tabs.

Example: In the splice process in Fig. 1(b), a context is defined for specific groups of operators. Only the users assigned to group *Wet-End* will receive information w.r.t. the splice process. Members of this group receive intrinsic, instance specific information like the quality of the paper roll to be used next. Since this is highly relevant data for executing task *T2*, this information is presented directly below the activity name. Furthermore, users receive the restmeters of paper on a specific roll (intrinsic) and the current glue temperature (extrinsic). Both values are numbers that are important information for operators but not directly related to activities. Thus, these values are visualized on additional tabs on the operators smartwatch. Figure 2 shows the characteristics in each dimension of the exemplarily described IoT variables.

Table 2. MQTT communication between IoT objects and BPMS

Topic	Description	Direction
$/\{variable\_id\}/data$	IoT sensor data	IoT to BPMS
$/\{actor\_id\}$	Device configuration	BPMS to IoT
$/\{actor\_id\}/tasks$	Tasks of specific actor	BPMS to IoT
$/\{actor\_id\}/\{variable\_id\}$	Context data for specific user	BPMS to IoT
$/\{actor\_id\}/command$	Actors actions (claim, complete, cancel)	IoT to BPMS
/keepalive		IoT to BPMS

4 Architecture and Implementation

The described approach has been implemented based on a three layer architecture that is visualised in Fig. 3. It consists of the following layers: *(i)* IoT objects, i.e., sensors as well as wearable devices of human participants; *(ii)* IoT infrastructure and communication middleware; and *(iii)* the BPMS. The layers are connected based on standard communication protocols. To allow the IoT objects at layer *(i)* to communicate with the IoT middleware at layer *(ii)* and the BPMS, respectively, a Message Queue Telemetry Transport (MQTT)[1] Broker is used. MQTT is a queue-based publish/subscribe protocol, which is especially suited for applications where computing power and bandwidth are constrained. The used MQTT topics are listed in Table 2. IoT objects, i.e., sensors or actuators, represent publishers. They are connected to an IoT gateway using specific architectures such as Profibus, LAN, WLAN or Bluetooth. A specific IoT variable v_x is acquired and published on a MQTT topic $/v_x/data$. Through a MQTT Broker the acquired data is sent to an acquisition application at layer *(ii)* that stores IoT data into a high performant NoSQL database that follows the database scheme described in Sect. 3.1. In our implementation we used the latest version of the Apache Cassandra database. A distribution application at layer *(ii)* keeps the BPMS updated with the latest IoT values. All running instances of a particular process receive the corresponding data value. The application cyclically acquires the values from the database in a Key-value structure and sends them to the BPMS. In our architecture we used the latest version of the Camunda BPMS and therefore communicated with the workflow engine by means of the Camunda Rest API[2], i.e., *PUT, POST* and *GET* HTTP requests as described in Fig. 3. The tools at layer *(ii)* ensure that process relevant information stemming from the IoT is up-to-date. Through the acquisition tool, IoT data meta information is provided that makes clear where the data stems from. Given the current IoT data values, the engine calculates available activities. As a mobile user interface we implemented an *Android* based smartwatch application that subscribes to specific MQTT topics. The distribution application cyclically requests the

[1] http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.
[2] https://docs.camunda.org/manual/7.8/reference/rest/.

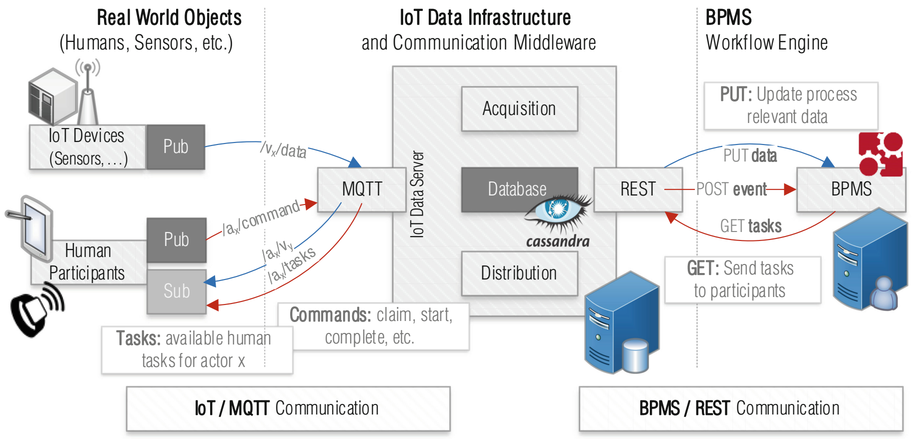


Fig. 3. Integrated communication architecture for IoT and BPMS

current user tasks from the Camunda API for each defined user and publishes to the correct MQTT topic, given the mobile device identifier, i.e., smartwatch device, configured on the BPMS (cf. Algorithm 1). The process of the device recognizing its configuration is implemented as follows: the distribution application cyclically checks the user configuration in the BPMS. When a change is detected, it publishes the new configuration to the topic $/\{actor\_id\}$. The smartwatch of a certain actor subscribes to the topic of its specific device identifier. Having established such connections, the smartwatch communicates with the MQTT broker by subscribing to the following topics: the current tasks for a specific operator are published on the topic $/\{actor\_id\}/tasks$. The device sends operators commands, such as *complete task* to the topic $/\{actor\_id\}/command$. The content of the message is forwarded straight to the BPMS using a *POST* request. Context-specific IoT data is sent to actors on topic $/\{actor\_id\}/\{variable\_id\}$ based on Algorithm 2. To prevent the MQTT service at the watch to be killed, we implemented a keep alive communication (topic = */keepalive*).

5 Evaluation and Industrial Application

In this section, we describe the evaluation of the proposed approach by means of an extensive application in corrugation industry. Due to increasing automation and staff reduction, less operators are available to control such a production line. Hence, interactions between users and machinery requires several location changes of users between control panels that result in delayed information flows. These delayed reaction times are frequently the reason for increased deficient products. The overall corrugation BPMN process model that is executed by a Camunda BPMS as described in Sect. 4 is visualised in Fig. 4. After initialising internal helper variables the control flow splits into different parallel paths

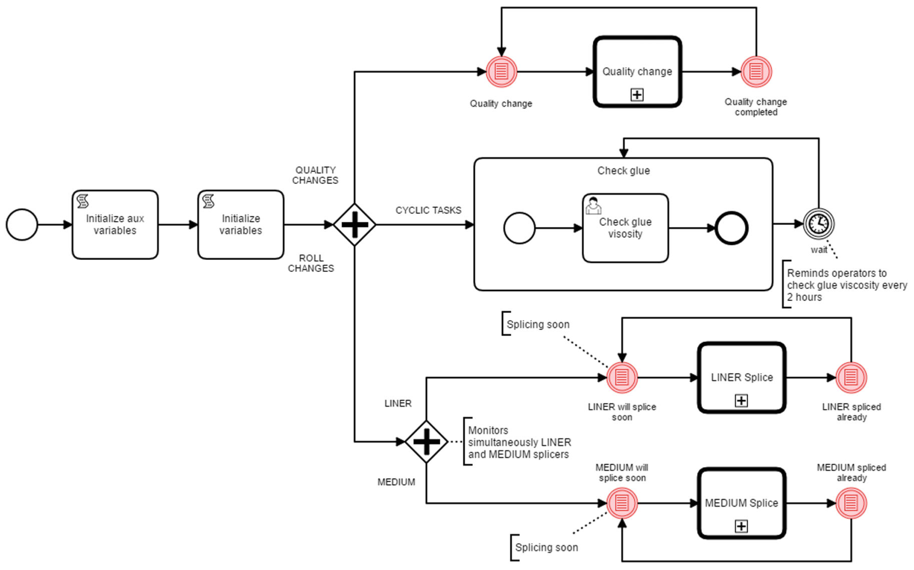


Fig. 4. Complete IoT enhanced process in corrugation industry (Color figure online)

where each path calls a specific subprocess depending on certain IoT based sensors conditions to be fulfilled (cf. red coloured Conditional Events). Each of these subprocesses reflects necessary operations that need to be performed to control production, e.g., the splice subprocess in Fig. 1. The implemented process contains all IoT enhanced modelling concepts that have been introduced in Sect. 3.2. To directly notify operators when human actions are needed, plant personal has been equipped with smartwatches, i.e., mobile process cockpits as described before and as shown in Fig. 5. Therefore, a user-group model has been defined in the BPMS. Here, available operators were assigned to a specific area of production that depicts their area of responsibility, e.g. machine *MF1*. Thus, depending on the area operators are working, the BPMS assigns a different set of tasks. Operators are then pointed to new human tasks through visual, acoustic and, in case of noisy environments through haptic signals like vibration alarms. Furthermore, operators are used more effective because low priority work is aborted in order to perform high priority work that could lead to machine stops.

In addition to currently available human tasks the IoT infrastructure provides diverse context-specific information on the smartwatch interface of operators. Depending on the specific group a user is assigned to, the wearable device offers diverse context information to operators: at the *Dry-End* (the area where produced corrugated paper leaves the plant), e.g., the remaining time of current production job, the remaining time to next stack transport, or the current production speed. Users at the *Wet-End* (the area where original paper is inducted to the plant) receive continuously information w.r.t. the next necessary roll change or occurring error and defects of machinery modules. In addition,

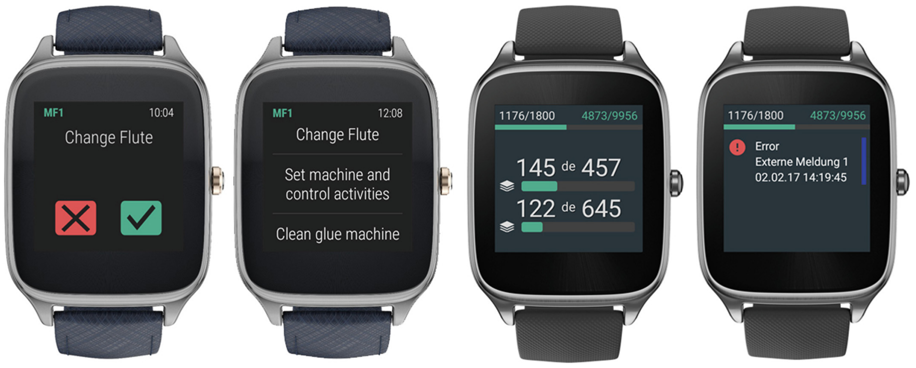


Fig. 5. Wearables: (a) unclaim/complete task; (b) tasks; (c) and (d) context info

operators receive error messages and environmental information from the different plant modules. This way, concrete and goal-oriented information in error cases or warning messages for supply shortfalls can be transmitted to operators and enhance the over all process transparency and thus the quality of task execution (cf. Challenges in Sect. 2.1). Through the described implementation it was possible to significantly reduce reaction time intervals. The amount of deficient products was decreased and the overall quality of the produced corrugated paper has been improved. The overall equipment downtime was significantly decreased, since problems have been prohibited or recognized in advance and were solved proactively. Hence, the overall equipment efficiency could be increased effectively. To quantify these findings, we analysed process execution with the Camunda Statistics Plugin[3]. We tracked the corrugation process *(i)* for five days without operators using wearable devices and *(ii)* other five days with operators being notified using smartwatches. In particular, we measured the average instance throughput time for a splice processes. The effectiveness of the approach has been measured based on machine stop times and waste reduction. On average, 100 splices are executed per shift, i.e., 8 h of production. In case *(i)* we recorded a total stop time of 180 min, i.e., on average 12 min per shift. In case *(ii)* the stop time has been decreased to 60 min in total, i.e., 4 min per shift on average. The results show that the application of the IoT enhanced BPMS leads to less machine stops because users need less time to recognize work to be done.

6 Related Work

Several approaches have been proposed to relate IoT objects with business processes. An overview of related approaches is given in Table 3. The table summarizes the support of each approach for IoT-aware process modelling, execution,

[3] https://github.com/camunda/camunda-cockpit-plugin-statistics.

Table 3. Overview on related approaches of the integration of IoT and BPM

Approach	Modelling	Execution	Monitoring	Mobile UI	Context
IAPMM [8,9]	✓	✗	✗	✗	✗
BPMN4CPS [10]	✓	✗	✗	✗	✗
BPMN for IoT [11–13]	✓	✗	✗	✗	✗
IoT/WS-BPEL [14]	∼ *(BPEL)*	✓	∼ *(BPEL)*	✗	∼ *(BPEL)*
IoT/WS-BPEL [15,17,18]	∼ *(BPEL)*	✓	∼ *(BPEL)*	✗	∼ *(BPEL)*
ADiWa[19]	∼ *(concept)*	∼ *(concept)*	✗	✗	✗
Extended-GSM [2–4]	✓	✗	✓*(via GSM)*	✗	✓
This work	✓	✓	∼ *(BPMS)*	✓	✓

and monitoring as well as the the availability of a mobile user interface and the possibility to provide (IoT) context information. In [8,9] the authors present the Internet-of-Things-Aware Process Modeling Method (*IAPMM*) that covers requirements analysis. It extends the BPMN meta-model to model IoT-aware processes. The approach in [10] (*BPMN4CPS*) also describes an extension of BPMN in which the process logic is split into the cyber part, the controller and the physical part. Furthermore the authors extended BPMN by new task types. Some more notation concepts in BPMN for IoT are described in [11–13]. The main focus is on the modeling of real world properties. Also in [11,12] the authors present an extension of BPMN with new modeling concepts. None of the described approaches provides details on how to execute these models. In [14] an approach for implementing an IoT-aware execution system given in WS-Business Process Execution Language (WS-BPEL) is introduced. It extends BPEL by *context variables* which are automatically updated. The authors implemented a prototype which is compliant with every WS-BPEL engine. Other approaches implementing BPEL extensions are presented in [15,16]. The variables are updated using the publish/subscribe paradigm. Another extension for WS-BPEL (*Context4BPEL*) with features to manage context events to allow the asynchronous reception of events, query context data and evaluate transition conditions based on context data is described in [17]. In [18] the authors integrate distributed resources into WS-BPEL by formalizing a fragment of WS-BPEL together with the WSRF (Web Services Resource Framework). In [19] the authors propose an approach for enabling IoT-based agile business processes. They provided concepts for extending models by triggers for variance. The approaches in [2–4] rely on the information coming from artifacts involved in the process to understand how a process evolves. By adopting an extension of Guard-Stage-Milestone (GSM), it is possible to monitor the process even when the control flow is not respected. The work presented in [20] introduces a lightweight process engine for enabling mobile applications. The authors describe requirements and concepts for mobile process applications and a prototypical mobile user interface. However, this work does not comprise IoT related aspects.

7 Conclusion and Future Work

In this paper, we introduced an integrated approach for IoT-aware business process execution that tackled several open challenges in this area of applied research. As a fundamental basis for the integration, we introduced an IoT data provenance framework. This technique allows us to consider current IoT data for interaction in arbitrary pre-defined process models that can be enacted by contemporary BPM execution systems. As demanded in many business cases, users need to be notified in real time when new tasks occur. This has been implemented by means of wearable user interfaces with configurable context specific IoT data provision. The approach has been implemented and evaluated extensively in production industry. For future research, we will focus on several IoT related issues in the area of BPM, e.g., the analysis of process execution event logs under consideration of IoT data.

References

1. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management, vol. 1. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
2. Meroni, G., et al.: Artifact-driven process monitoring: dynamically binding real-world objects to running processes. In: CAiSE Forum (2017)
3. Meroni, G., Di Ciccio, C., Mendling, J.: An artifact-driven approach to monitor business processes through real-world objects. In: Maximilien, M., Vallecillo, A., Wang, J., Oriol, M. (eds.) ICSOC 2017. LNCS, vol. 10601, pp. 297–313. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69035-3_21
4. Meroni, G., et al.: Multi-party business process compliance monitoring through IoT-enabled artifacts. Inf. Syst. **73**, 61–78 (2018)
5. Janiesch, C., Koschmider, A., et al.: The Internet-of-Things Meets Business Process Management: Mutual Benefits and Challenges. CoRR, vol. abs/1709.03628 (2017)
6. Schönig, S., Jablonski, S., Ermer, A., Aires, A.P.: Digital connected production: wearable manufacturing information systems. In: Debruyne, C., Panetto, H., Weichhart, G., Bollen, P., Ciuciu, I., Vidal, M.-E., Meersman, R. (eds.) OTM 2017. LNCS, vol. 10697, pp. 56–65. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73805-5_6
7. Mahdi, R., Jablonski, S., Schönig, S.: Extrinsic dependencies in business process management systems. In: ICEIS (2018)
8. Petrasch, R., Hentschke, R.: Process modeling for industry 4.0 applications - towards an industry 4.0 process modeling language and method. In: International Joint Conference on Computer Science and Software Engineering (2016)
9. Petrasch, R., Hentschke, R.: Towards an internet-of-things-aware process modeling method - an example for a house suveillance system. In: Management and Innovation Technology International Conference (2015)
10. Graja, I., et al.: BPMN4CPS: a BPMN extension for modeling cyber-physical systems. In: WETICE, pp. 152–157. IEEE (2016)
11. Meyer, S., Ruppen, A., Magerkurth, C.: Internet of things-aware process modeling: integrating IoT devices as business process resources. In: Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) CAiSE 2013. LNCS, vol. 7908, pp. 84–98. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38709-8_6

12. Meyer, S., Ruppen, A., Hilty, L.: The things of the internet of things in BPMN. In: Persson, A., Stirna, J. (eds.) CAiSE 2015. LNBIP, vol. 215, pp. 285–297. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19243-7_27
13. Sperner, K., Meyer, S., Magerkurth, C.: Introducing entity-based concepts to business process modeling. In: Dijkman, R., Hofstetter, J., Koehler, J. (eds.) BPMN 2011. LNBIP, vol. 95, pp. 166–171. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25160-3_17
14. Domingos, D., Martins, F., Cândido, C., Martinho, R.: Internet of things aware WS-BPEL business processes context variables and expected exceptions. J. Univ. Comput. Sci. **20**(8), 1109–1129 (2014)
15. George, A.A.: Providing context in WS-BPEL processes. Master's thesis, University of Waterloo (2008)
16. George, A.A., Ward, P.A.: An architecture for providing context in WS-BPEL processes. In: Conference of Advanced Studies on Collaborative Research (2008)
17. Wieland, M., Kopp, O., Nicklas, D., Leymann, F.: Towards context-aware workflows. In: CAiSE Workshops and Doctoral Consortium, vol. 2, p. 25 (2007)
18. Mateo, J.A., Valero, V., Dıaz, G.: BPEL-RF: a formal framework for BPEL orchestrations integrating distributed resources, preprint arXiv:1203.1760 (2012)
19. Schmidt, B., Schief, M.: Towards agile business processes based on the Internet of Things. In: Advanced Manufacturing and Sustainable Logistics, pp. 257–262 (2010)
20. Schobel, J., Pryss, R., Schickler, M., Reichert, M.: A lightweight process engine for enabling advanced mobile applications. In: Debruyne, C., Panetto, H., Meersman, R., Dillon, T., Kühn, E., O'Sullivan, D., Ardagna, C.A. (eds.) OTM 2016. LNCS, vol. 10033, pp. 552–569. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_33

Flexibility in Business Process Modeling to Deal with Context-Awareness in Business Process Reengineering Projects

Leila Jamel[1(✉)], Oumaima Saidani[1], and Selmin Nurcan[2,3]

[1] Information Systems Department, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia
{LMJamel,OCSaidani}@pnu.edu.sa

[2] Université Paris 1, Panthéon-Sorbonne Centre de Recherche en Informatique, 90, rue de Tolbiac, 75634 Paris Cedex 13, France
Selmin.Nurcan@univ-paris1.fr

[3] IAE de Paris – Sorbonne Graduate Business School, 21, rue Broca, 75005 Paris, France

Abstract. Current research on Business Process Management (BPM) outlines the importance of Business Process Reengineering (BPR) operations for business improvements' goals. One of the main issues when conducting this kind of project is basically related to the modeling of current (as-is) and future (to-be) processes in the enterprise. In fact, traditional business process modeling techniques (BPMT) are being difficult to adopt for BPR projects which are being more and more context-aware, especially for two root causes: (i) BPR projects are conducted in a changing environment and (i) Business Process (BP) requirements are constantly evolving especially within permanent customers' changing needs. Thereby, the contexts in which BPR projects are conducted, are different; they may have various purposes, may cover different perspectives of the organization, etc. That's why business process modelling techniques supporting flexibility could be more suitable to deal with these changing atmosphere. In this paper, we propose the use of an approach for flexible business process modeling BPVM [1] for BPR projects.

Keywords: Business process reengineering
Business process modeling techniques · Flexibility · Context-awareness

1 Introduction

Many researches on Business Process Management (BPM) highlight the importance of Business Process Reengineering (BPR) operations for business improvements' goals allowing them to better support current business requirements.

In the current complex and dynamic environments, Business Processes (BP) have to be flexible and adaptable to changes in theses environments. Flexibility in business process modeling has been the focus of many researches [24, 25, 30, 31]. There are many definitions of the flexibility in literature. It is defined in [32] as "the ability to yield to change without disappearing". We define flexibility as the capacity of making a compromise between, first, satisfying, rapidly and easily, the business requirements in terms of adaptability when organizational, functional and/or operational changes occur; and, second, keeping effectiveness. Among techniques used for dealing with flexibility we mention (i) *variability* which iss defined as "the ability of deriving different variants from the same BP" [31], (ii) *adaptability* that can be defined as the ability to temporarily deviate the flow during the execution of a BP [31], and (iii) *context-awareness* which is the ability to use contextual information to adapt the process [12]. A lot of approaches dealing with techniques implementing flexibility were proposed. While some of them focus on context-awareness and adaptation [12, 19, 26], others propose reuse and variability mechanisms [1, 2, 8, 21, 27–29].

Moreover, the needs of approaches to deal with context-awareness in BPR projects are significant. Many researches in business process management stress the importance of context-awareness issues in different aspects related to BPR. In fact, BPR projects are often context dependent; not only the environment in which BPR projects are lead is changing, but also, business requirements are evolving. More precisely BPR projects can have different purposes and may cover different modeling perspectives in the organization.

Furthermore, one of the main phases when conducting BPR projects is related to the modeling of current (as-is) and future (to-be) processes in the organization. These tasks need the use of business process modeling techniques (BPMT).

Adopting a non-flexible BPMT in a BPR project, which being context-aware, may makes difficult the realization of these projects.

That is why, we argue that business process modelling techniques supporting flexibility could be more suitable to deal with these changing atmosphere.

In this paper, we propose the use of an approach for flexible business process modeling BPVM [1] to model as-is and to-be business processes in BPR projects. We focus on the BP design and redesign.

The rest of this paper is organized as follows: Sect. 2 presents an overview of BPR and introduce a set of research issues for dealing with context-awareness in this field, and particularly in the phases related to design steps in BPR life cycle. Section 3, presents a flexible business process modeling approach for the formalization of existing business processes to redesign (as-is), for the identification of the changes, and for the modeling of future business processes (to-be) in the organization with a flexible way. This approach is called BPVM (Business Process Variability Model).

2 BPR (Business Process Reengineering) and Context-Awareness

According to Hammer et al., BP renginering is the fundamental rethinking and radical redesign of business processes to achieve dramatic improvements in critical, contemporary measures of performance such as cost, quality, service and speed [5]. Also, BPR has been used by organizations as an approach to implement and manage changes [9]. A key issue in BPR is the 'how' question. BPR requires some methodology guidelines [3, 4].

The Business process reengineering (BPR) is one of the most adopted methods to introduce change in enterprises. In fact organizations often have to change their processes at higher or lower frequencies, in order to improve and make them more efficient. To reach this goal a BP modelling task is to be fulfilled during the analysis, diagnose and redesign BPR phases.

Modeling business processes is very helpful for the success of BPR projects according to the several advantages it offers, such:

– understanding the existing BPs, identifying their weakness and problems,
– identifying areas of potential improvement and areas with a gap between existing BPs and the BPR objectives,
– representing new BPs in order to evaluate their performance
– increasing the speed and the quality of the implementation of BPR improvements
– being used for end-user training: all documents such as work instructions, user instructions, ISO documents, etc. are stored in the model that constitutes a single information source
– being used as "the best practices models", BPs models can be used as start point in similar companies initiating BPR projects.

The modeling task is not easy to be put into practice in BPR projects due to the huge number of emerging business process modeling techniques, notations and languages (e.g. Event-driven Process Chains (EPC), UML Activity Diagrams, Business Process Modeling Notation (BPMN) [11], Business Process Execution Language (BPEL), etc.) [3]. A lot of researches dealing with this problem were performed [10, 15, 16, 20].

Some research labors are based on the track of context awareness in BPR to solve this difficulty. Among research works dealing with context aware BPR we mention [3, 4, 7].

The authors of [4] introduce a Context-aware Business Process Evaluation and Redesign approach which consists in two principle steps: first, it identifies the BP context; second, it enhances the original BP model by using process chunks appropriate for this context. The proposed approach use the workflow patterns [6] as an extension mechanism in the phase of redesign. To each pattern is associated a relevance degree in a particular context of use.

In [3, 7] authors propose a metamodel for BPR vocations which encompasses knowledges and concepts related to (i) modeling BPs in the context of the enterprise and (ii) BPs reengineering. In fact a guidance approach is performed for selecting adequate BP models to be used for as-is or to-be BPs during BPR projects. This approach is based on a classification framework for about 30 traditional BP modelling techniques (e.g. EPC, Petri nets, UML class diagrams, UML activity diagrams, IDEF 0, State/transition diagrams, BPMN, Role Activity diagrams, etc.). The framework is built on four dimensions: (i) Type of business processes, (ii) Degree of change, (iii) Modeling purpose, and (iv) Modeling view.

(i) Type of business processes: which reflects the three ISO classification of the enterprise BPs: Core, Management and Support BPs [33].

  – Core (operational) processes: are the operational processes of the business which result in the production of the outputs that are required by external customers,

- Support processes: are those that enable the core processes to exist. They concentrate on satisfying internal customers,
- Management Processes: concern themselves with managing the core processes or the support processes, or they concern themselves with planning at the business level.

(ii) Degree of change (level of change): which reflects the degree (or the scale) of BPR radicalness. In fact, a BPR project might consider one or many BP at one time and, according to fixed objectives. For each BP a level of change is then defined. In fact a BPMT might be more appropriate than others to reach a fixed change level. Levels of radicalness are: radical and incremental (Fig. 1). So if the level is radical then the BPMT should allow the redesign of the BP, i.e. it is necessary to remodel the process in order to have radical changes. In contrast, if the level is incremental then the BPMT should allow modelers to interact with the obtained BP model in order to bring continuously and incrementally the desired changes [3, 7].

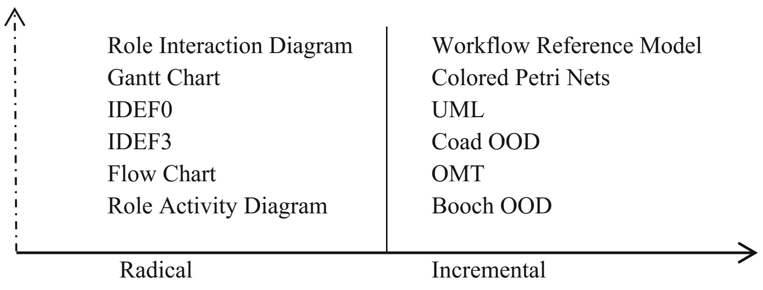


Fig. 1. Classification of BPMTs based on change degree.

(iii) Modeling purpose: which reflects objectives reached from the processes modeling. In fact in BPR actions it is a question of defining new BPs or redesign existing ones. Theses actions require BPMT for different objectives which might be: analysis, decision, decision execution and information technology solutions support [3, 7]. Figure 2 presents a classification of some traditional BPMTs according to modeling purposes.

(iv) Modeling view (modelling perspective): a BP model should be capable of providing the necessary BP information elements, such as: what are the activities composing the process, who is performing these activities, what elements they provide, where and how, etc. So a BPMT supports many modeling views (Fig. 3), such as: functional (what), behavioral (when and how), informational (entities produced by or manipulated by a process) and organizational (where and by which role).

Even if the introduced framework was proposed to help analyzing and evaluating business process modeling techniques, it can also be used for the definition of new

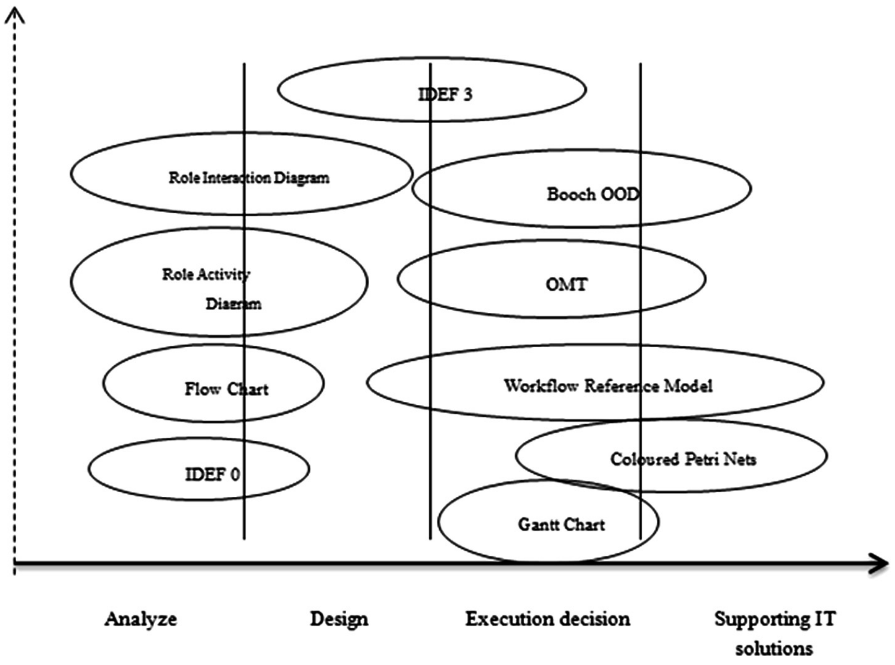


Fig. 2. Classification of BPMTs based on modeling purposes [3, 7]

ones. That is why, is Sect. 3, we will introduce BPVM as a modeling technique, supporting flexibility, for BPR projects.

3 BPVM (Business Process Variability Model) for Context-Aware BPR Projects

We introduce in this section the business process modeling approach for the representation of flexible business processes BPVM [17] and we demonstrate its adequacy as a modeling technique in BPR projects. We start by presenting an overview and the meta-model of BPVM in Sect. 3.1; then, in Sect. 3.2, we describe briefly the main concepts of this approach while situating it with reference to the multidimensional classification structure proposed in [3, 7]. We demonstrate the manner it embodies the capability to take into consideration flexibility requirements and support most of the dimensions introduced in the framework introduced in Sect. 2.

3.1 Overview of BPVM

BPVM is a multi-perspectives business process modeling approach integrating variability. It allows to represent business processes in a way to give them their capacity to be adaptable, on the one hand, and to identify and to formalize the factors whose variations require changes at run-time (i.e. context, and quality requirements), on the

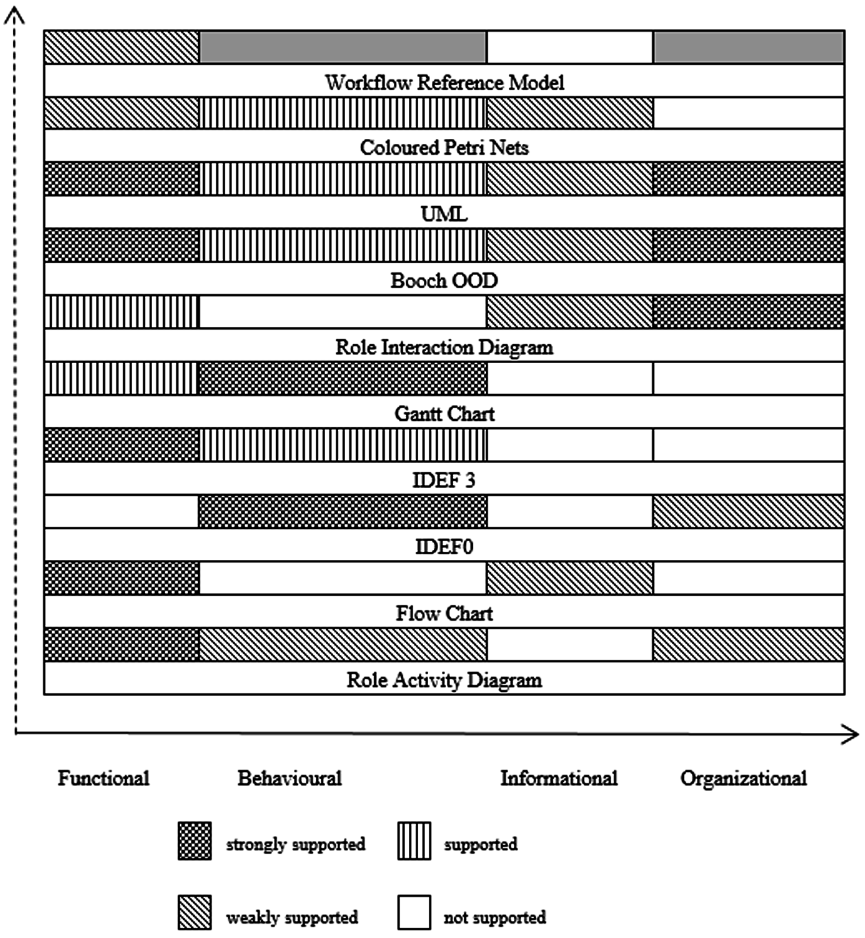


Fig. 3. Classification of BPMTs based on modeling perspectives [3, 7]

second hand. The proposed approach allows to build several possible representations of a business process and to capture change requirements that affect the process execution.

Figure 4 shows the meta-model BPVM using the notation of UML class diagram. The proposed meta-model includes five parts that cover the following perspectives: the intentional perspective, the functional perspective, the organizational perspective, the non-functional perspective and the non-organizational resource perspective. As shown in Fig. 4, the core concept in BPVM is that of business process fragment (BPF). The perspectives of the meta-model are interconnected through this concept.

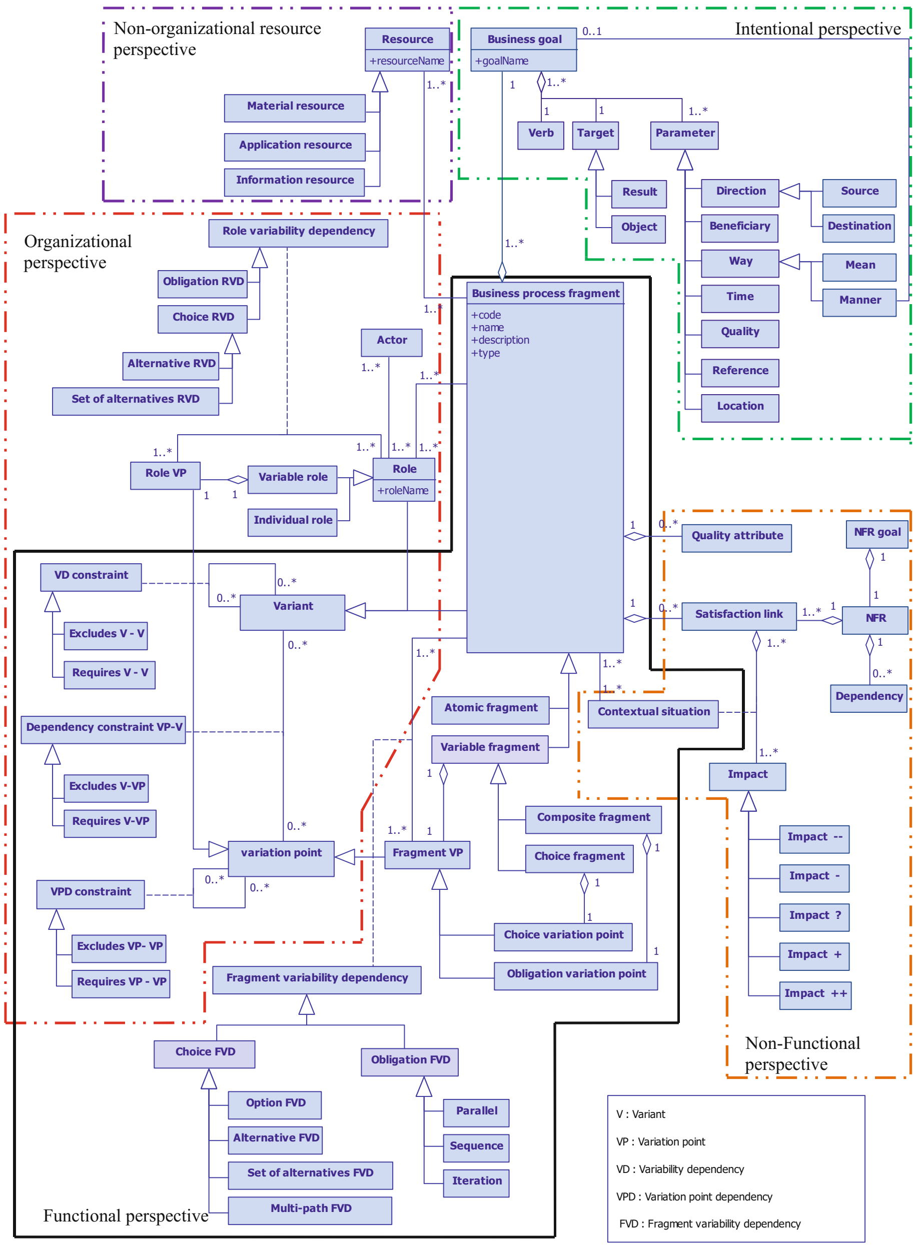


Fig. 4. BPVM Meta-model [17]

3.2 Analysis of BPVM with Reference to the Modeling Techniques Classification Framework

In this section, we demonstrate the adequacy of BPVM for modeling BP in context-aware BPR by analyzing it with reference to the dimensions of the classification framework introduced in Sect. 2.

a. **Dimension 1: Modeling view**

To represent flexible processes, a modeling solution has to provide a minimum set of perspectives to represent the enterprise elements that are potentially impacted by changes [24].

Regarding the dimension *Modeling view*, BPVM is a multi-perspectives business process modeling approach which allows business processes to be represented according to five perspectives: *intentional perspective*, *functional perspective*, *non-functional perspective*, *organizational perspective* and *non-organizational resources* perspective.

The functional view. The functional view represents the BPFs by specifying their functional composition of units of finer granularity. This composition follows a hierarchical structure whose leaves fragments represent atomic processes. This perspective represents a business process model in terms of BPFs which have to be achieved as well as their structures, the composition links and the variability dependencies between them, and the conditions and the constraints governing their achievements.

A BPF is defined as a part of a business process model that (i) creates value for the organization, (ii) can be reused in several process models, (iii) can be placed under the responsibility of one or more roles (iv) and whose implementation allows to satisfy a business goal. This concept aims to define multiple levels of abstraction. It is similar to the concept of sub-process defined by the WfMC [10] and the OMG [11].

The Organizational View. This view allows to express the organizational resources which are required for the business process realization. These resources are the actors and the roles they play. The core concept in this perspective is that of role. A role is defined as an organizational entity which is responsible for the achievement of a BPF and that can be assigned to one or more actors; it can represent a skill, a competency or qualification or a responsibility. It can also represent a group of individuals. The concept of role is also considered as a means allowing to assign the actors to the BPFs instances. This concept is similar to the concepts of business role and business entity defined in BPMN(Business Process Modelling Notation), to the concept of organizational unit defined in EPC, and to the concept of organizational role defined by the WfMC [13]. An actor is defined as a resource that is involved in the execution of a process instance fragment since it is assigned to a role responsible for the achievement of this fragment. An actor is assigned to one or more roles based on their qualifications and skills. An actor may be responsible for the achievement of one or more instances of BPFs according to the roles they can play. This concept is similar to that of participant defined by the WfMC.

The Behavioral View. BPFs define the structure of a process and they can cover the following modeling situations: atomicity, composition, sequence, parallelism, optionality and choice (alternative or multiple).

A fragment variation point is a representation of one or more places to which an obligation of selection or a choice decision is attached. The choice decision is made based on the intention of the actor, the context, the responsible role and the desired quality properties. Each variable BPF is associated to a fragment variation point.

The association of a BPF to a variation point is expressed using the relationship *Fragment variability dependency* (FVD). Figure 4 shows two types of FVD: *obligation* and *choice*. An obligation FVD can be of three kinds: *parallel*, *sequence* or *iteration*. A choice FVD can be of four types: option, alternative, set of alternatives or path.

The Non-functional View. This view formalizes the non-functional requirements that a business process have to meet and the qualitative goals of the organization which allow improving the quality of the business processes [18, 22]. As shown in Fig. 4, BPVM expresses the quality requirements related to business processes as well as the satisfaction links between the goals and the BPFs, and the impact values according to a given context. “Accuracy” and “Safety” are examples of quality requirements. The information about the impact of a non-functional requirement (NFR) on every fragment is considered as a quality attribute for this fragment. Thus, the quality of the business process is expressed through the quality of its components, i.e. the BPFs. As shown in Fig. 4, the quality of a BPF is formalized by the use of the links between the classes “Business process fragment” and respectively the classes “Quality attribute” and “Satisfaction link”.

The Informational View. The informational view is supported by the non-organizational resources perspective in BPVM. Non organizational resources are the resources used - or produced - by process fragments: data, business objects, and so on. Resources can be of different natures. As shown in Fig. 4, three subclasses of resources are distinguished: information resources, application resources, and hardware resources. A resource can be used by a process fragment. It can also be “consumed” (for some hardware resources), or produced during the execution of the process.

The Intentional View. The intentional perspective allows expressing the goals that business processes have to meet. It is represented, in the meta-model by the fact that a BPF (Business Process Fragment) achieves a goal. The relationship between BPFs and goals which can be achieved by these BPFs is formalized by the link between the classes *Business process fragment* and *Business goal*. In BPVM, a business goal specifies an objective that we have to achieve without detailing how to achieve it. It identifies the needs and the expectations attached to a business process. A business goal is defined as an objective of the organization in carrying out its activities which is satisfied through the realization of one or several BPFs. Business goals are formalized using a linguistic approach that is based on the formalism proposed in [23]. This formalism provides a support for the business processes engineering based on goals [5]. The linguistic template of a goal includes a verb, a target and a set of parameters that play specific roles related to the verb. The list of parameters is as follows: *Source* and *Destination* (which are generalized by the parameter *Direction*), *Means* and *Manner* (which are generalized by the parameter *Way*), *Beneficiary*, *Time*, *Quality*, *Reference* and *Location*. The verb and the target are mandatory, whereas the parameters are optional. The target designates the entity affected by the goal. It can be of two kinds: *object* or *result*. The parameters are defined in details in [17].

The Contextual View. The contextualization of a business process model (obtained by the instantiation of BPVM) consists in informing all its conditions of applicability of the BPFs. This is done by representing the context characteristics and the contextual conditions. Two kinds of contextualization are proposed: the functional contextualization and the non-functional contextualization.

The first kind of contextualization consists in expressing the contextual conditions related to BPFs and to the roles and in representing the impact of the context on the way of executing these BPFs and of choosing the appropriate BPFs and roles at run-time. A contextual condition is associated to every BPF allowing to specify the conditions under which the execution of the BPF is possible.

The non-functional contextualization consists in adding the contextual conditions to the quality attributes. In fact, in some situations, the context has an impact on the contribution value of the variants in the satisfaction of a quality goal, i.e. according to the context, and according to the desired quality purposes, it is better to select an alternative rather than another one.

b. **Dimension 2: Modeling purpose**

In BP reengineering projects, BPVM can be used not only for defining new BPs, but also for or redesigning existing ones. Moreover, it is adequate to use for many modeling purposes which might be: analysis, design, and decision execution. Firstly, BPVM can define business processes at the intentional level as well as et the operational level; secondly, regardless of the abstraction level, using the concept of BP fragment different levels of abstraction can be expressed. In fact, BPVM allows to represent high-level orientation views of business processes, aiming to understand business environment, defining business decisions, strategies and goals.; also more detailed views of the business activities can be expressed, as well, the diffrent aspects related to business activities (i.e. the behaviour, resources, etc…).

c. **Dimension 3: Types of processes**

BPVM can be used for the representation of (i) core processes which represent the essential activities the organization whose achievement allows to satisfy organizations goals and objectives, (ii) management processes that are designed to plan, to monitor and to control business activities, and (iii) support processes which assist the value-delivering core processes by providing the resources and infrastructure required by them.

d. **Dimension 4: Degree of change**

BPVM can be used for radical change and incremental change. It deals with the needs related to the reuse and the modularity. The concept of BPF that we propose allows to define modular and reusable components which are linked to goals to satisfy. BPVM supports variability in the organizational and the functional perspectives.

Functional Variability. Using BPVM, a business process model can be represented by a set of BPFs which can be achieved in different contexts and by different actors that can have various preferences on the manner in which their intentions are achieved. Thus, a BPF can be achieved in different ways and different alternatives for the accomplishment of a business process model can be defined.

In order to deal with variability, BPVM introduces key concepts related to variability: *variation point* and *variant* which are based on OVM (Orthogonal Variability Model) [14]. In BPVM, BPFs and roles are considered as variability units. BPVM extends OVM by the concepts of *role* and *BPF*. These two concepts refer to the concept of variant in OVM. Figure 4 shows the meta-model of OVM extended by the concepts of BPF and role (which specializes the concept of variant in the original model) as well as the concepts of *variation point role* and *variation point fragment*. According to the meta-model, a variation point is a point in the business process where a change occurs indicating the existence of various realization alternatives. A variant is a possible alternative related to a variation point. The variants and the variation points are connected by *variability dependencies*. The variability dependencies can be of two types: *choice* and *obligation*. As shown in Fig. 4, a set of dependency constraints between the variants, between the variation points, and between the variants and the variation points, are defined. The dependency constraints represent the rules that have to be followed to ensure the consistency of the business process instances. BPVM distinguish two types of dependency constraints similar to those defined by FODA (Feature Oriented Domain Analysis): the *Requires* and the *Excludes* constraints.

Organizational Variability. The organizational perspective expresses the variability dependencies between the roles. Like the dependencies of variability between process fragments, the dependencies between the roles are based on the variability model OVM. As shown in Fig. 4, two kinds of roles are defined: individual role and variable role.

A BPF can be achieved under the responsibility of several actors playing different roles. The concept of variable role is used. Roles and variability dependencies between them constitute a role hierarchy whose leaves represent individual roles. The purpose of this representation is to provide a mechanism for flexible assignment of the BPFs to the actors playing various roles. Thus, the same BPF can be achieved by different roles in different situations.

An individual role is a role that does not include other roles. Director is an example of individual role. A variable role is an entity that expresses an organizational variability by grouping a set of roles. We identify three kinds of variables roles: (i) composite role which consists of two or more roles, (ii) alternative role which includes mutually exclusive roles and (iii) set of alternatives-roles which includes a set of roles from which at least one role is selected at run-time. A variation point is associated to each variable role.

Thus, we can resume our BPVM, according to the multidimensional classification structure for BPM techniques presented in Sect. 2, with extending the mentioned framework with the intentional view, by defining the different dimensions and charecteristics proposed, as illustrated in Table 1. We note that we have extended the classification framework by three views in the dimension modeling view. These views are the contextual view, the intentional view and the non-functional view.

Table 1. Analysis of the BPVM according to a set of dimensions from the extended modeling techniques selection framework

Dimension	Value	Support
Modeling view	Informational view	*Yes*
	Functional view	*Yes*
	Organizational view	*Yes*
	Behavioral view	*Yes*
	Intentional view	*Yes*
	Non-functional view	*Yes*
	Contextual view	*Yes*
Modeling purpose	Analysis	*Yes*
	Design	*Yes*
	IT solution support	*Yes*
Type of process	Management processes	*Yes*
	Core (operational) processes	*Yes*
	Support processes	*Yes*
Degree of change	Radical	*Yes*
	Incremental	*Yes*

4 Conclusion

In this paper, we discussed the use of a business process modeling approach (BPVM) which supports flexibility, for modeling the "as-is" and the "to-be" business processes in context-aware BP reengineering projects. This approach allows to represent a business process model according to many modeling perspectives. Also, it can be used to model different types of business processes. What's more, it can be used for analysis or design purposes. And finally, it allows to take into consideration both radical and incremental change; it fact it supports variability in both the functional and the organizational perspectives hence variations are defined with respect to the way of achievement of business process fragments and to the actors' roles.

The work presented in this paper in founded on a multidimensional classification framework of many used traditional BPMT which presented the goal of helping analyzing and evaluating business process modeling techniques and consequently the definition of new ones.

In the future, we will continue our efforts to evaluate the use of BPVM in BPR projects on a large BPs panoply. It will be interesting to work on software development processes which funded on COTS (Commercial off-the-shelf) software components as they are in a close relationship with the context in which they are chosen, applied, tested and maintained by software developers.

References

1. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. Inf. Syst. **32**(1), 1–23 (2007)
2. Korherr, B.: Business Process Modelling. Languages, Goals and variabilities. VDM Verlag (2013)
3. Ayachi, S. Jamel Menzli, L.: Proposition et expérimentation d'un métamodèle pour la réingénierie des processus métiers. ISI, vol. 13/3, pp. 105–129 (2008)
4. Bessai, K., Claudepierre, B., Saidani, O., Nurcan, S.: Context-aware business process evaluation and redesign. In: Workshop on Business Process Modelling, Development, and Support (BPMDS), Montpellier, France, June 2008
5. Hammer, M., Champy, J.: Reengineering the Corporation: A Manifesto for Business Revolution. Harper Collins, London (1993)
6. van der Aalst, W.M.P., Alignement, B.: Using process mning as a tool for delta analysis and conformance testing. Requir. Eng. J. **10**(3), 198–211 (2005)
7. Jamel Menzli, L., Ayachi Ghannouchi, S., Hajjami Ben Ghézala, H.: A guidance process for the selection of business process modelling techniques for the revised business process reengineering. JCIT **2**(2), 79–88 (2007)
8. Rolland, C., Prakash, N.: On the adequate modeling of business process families. In: The 8th Workshop on Business Process Modeling, Development, and Support (BPMDS 2007), Held in Conjunction with CAiSE 2007 (2007)
9. Goksoy, A., Ozsoy, B., Vayvay, O.: Business process reengineering: strategic tool for managing organizational change an application in a multinational company. Int. J. Bus. Manag. **7**(2), 89–112 (2012)
10. Röglinger, M., Pöppelbuß, J., Becker, J.: Maturity models in business process management. Bus. Process Manag. J. **18**(2), 328–346 (2012)
11. Business Process Model and Notation (BPMN), Version 2.0, Document number formal/2011-01-03, (2011). http://www.omg.org/spec/BPMN/2.0
12. Saidani, O., Rolland, C., Nurcan, S.: Towards a generic context model for BPM. In: Hawaii International Conference on System Sciences (HICSS 2015), pp. 4120–4129 (2015)
13. Workflow Management Coalition Specification: Workflow Management Coalition Terminology and Glossary. Technical report Number WFMC-TC-1011 (1999)
14. Pohl, K., Böckle, G., van der Linden, F.: Software Product Line Engineering Foundations, Principles and Techniques. Springer, Heidelberg (2005). https://doi.org/10.1007/3-540-28901-1
15. Adamides, E.D., Karacapilidis, N.: A knowledge centred framework for collaborative business process modelling. Bus. Process Manag. J. **12**(5), 557–575 (2006)
16. Teplov, V.I., Tarasova, E.E., Matuzenk, E.V., Alyabieva, M.V., Belenov, E.O.: Commercial activity business processes reengineering: theoretical, methodological and practical aspects. J. Adv. Res. Law Econ **VII**(3(17)), 649–661 (2016)
17. Saidani, O., Nurcan, S.: Business process modeling: a multi-perspective approach integrating variability. In: Bider, I., Gaaloul, K., Krogstie, J., Nurcan, S., Proper, Henderik A., Schmidt, R., Soffer, P. (eds.) BPMDS/EMMSAD -2014. LNBIP, vol. 175, pp. 169–183. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43745-2_12
18. Chung, L., Nixon, B.A., Yu, E.: Dealing with change: an approach using non-functional requirements. Requir. Eng. J. **1**, 238–260 (1996). Proceedings of the Second Internation Symposium on Requirements Engineering. Springer Verlag London Limited, York

19. Ploesser, K., Recker, J., Rosemann, M.: Challenges in the context-aware management of business processes: a multiple case study. In: 19th European Conference on Information Systems (ECIS), Helsinki, Finland (2011)
20. Wenhong, L., Tung, A.Y.: A framework for selecting business process modeling methods. Ind. Manag. Data Syst. **99**(7), 312–319 (1999)
21. La Rosa, M., Dumas, M., ter Hofstede, A.H.M., Mendling, J.: Configurable multi-perspective business process models. Inf. Syst. **36**(2), 313–340 (2011)
22. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Academic Publishers, Boston (2000)
23. Prat, N.: Aide au processus de BPR: une approche fondée sur le raisonnement par cas. In: Colloque de l''Association Information et Management (AIM), Cergy-Pontoise, France, pp. 212–227 (1999)
24. Nurcan, S.: A survey on the flexibility requirements related to business processes and modeling artifacts. In: Proceedings of the 41st Annual Hawaii International Conference on System Sciences, Waikoloa, HI, USA, pp. 7–10, January 2008
25. Saidani, O., Nurcan, S.: A role-based approach for modelling flexible business processes. In: The 7th Workshop on Business Process Modelling, Development, and Support (BPMDS 2006), Luxembourg (2006)
26. Saidani, O., Nurcan, S.: Towards context aware business process modelling. In: The 8th Workshop on Business Process Modelling, Development, and Support (BPMDS 2007), Trondheim, Norway (2007)
27. Ayora, C., Torres, V., Weber, B., Reichert, M., Pelechano, V.: VIVACE: a framework for the systematic evaluation of variability support in process-aware information systems. Inf. Soft. Technol. **57**, 248–276 (2015)
28. Mechrez, I., Reinhartz-Berger, I.: Modeling design-time variability in business processes: existing support and deficiencies. In: Bider, I., et al. (eds.) Enterprise, Business-Process and Information Systems Modeling. Lecture Notes in Business Information Processing, vol. 175, pp. 378–392. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43745-2_26
29. La Rosa, M., van der Aalst, W.M., Milani, F.P., Dumas, M.: Business process variability modeling: a survey. ACM Comput. Surv. **50**(1) (2013). Article No. 2
30. Murguzur, A., Intxausti, K., Urbieta, A., Trujillo, S., Sagardui, G.: Process flexibility in service orchestration: a systematic literature review. Int. J. Coop. Inf. Syst. **23**(3), 343–374 (2014)
31. Cognini, R., Corradini, F., Gnesi, S., Polini, A., Re, B.: Business process flexibility - a systematic literature review with a software systems perspective. Inf. Syst. Front. J. Res. Innov. **20**, 343–371 (2018). ISSN: 1387-3326 (Print) 1572-9419
32. Regev, G., Wegmann, A.: A Regulation-based view on business process and supporting system flexibility. In: Proceedings of the CAiSE 2005 Workshop, pp. 91–98 (2005)
33. Harmon, P.: Business Process Change: A Business Process Management Guide for Managers and Process Professionals, 3rd edn. Morgan/Kaufmann Publishing, Boston (2014)

Business Process Canvas as a Process Model in a Nutshell

Georgios Koutsopoulos and Ilia Bider(✉)

DSV, Stockholm University, Stockholm, Sweden
{georgios,ilia}@dsv.su.se

Abstract. The paper suggests using a business process canvas as a model of a process in a nutshell that presents the essential properties of the process and the context in which it is run, including the position of the process in the business process ecosystem. The canvas consists of three sections: positioning, operations and resources. A positioning section, called Outside, includes such components, as the purpose of the process existence, strategic goals (to be) achieved by having the process, related processes and mechanisms of initiation of new process instances. The operations sections, called Inside, gives an overview of the work of the process instance, and it includes such components as operational goal, milestones, main events and activities, outcomes and constraints. The resources section, called Resources, describes resources/assets used in the process instances, and includes such components, as participants, tools, methods, etc. The paper proposes a canvas layout, describes its components, and presents an example. In conclusion, the paper discusses areas where the canvas could be used in practice.

Keywords: BPM · Business process · Business process modeling
Canvas · Goal · Context

1 Introduction

Patric Hoverstadt, in [1] writes "It is now very common to find organizations where as soon as you scratch the surface you discover that nobody really understands how the whole business works". In the same article, he also highlights "the need for an architectural model that provides business leaders with a model of the enterprise that they can genuinely use to understand how it operates". We believe that these reflections are valid not only for the whole business, but also for a complex business process, especially if it stretches over the whole enterprise/organization and crosses the boundaries of departments and teams. Any decision related to a particular process, e.g. redesign, optimization, introducing a new IT system to support the process, requires understanding of the process. This understanding includes not only particular details of how it is currently run internally, but also why it exists and how it is related to other processes, i.e. its position in the business processes ecosystem.

There are many ways to model a business processes, such as workflow modeling [2], state-oriented modeling [3], data-centered and artifact-centered modeling, etc. However, all these modeling techniques concern representing the particulars of internal behavior of the process instances, and do not depict all relevant details of the context. Besides, all these techniques use complicated diagrammatic languages that make it difficult to grasp the main characteristic of the process. As far as relationships between the business processes are concerned, there are a number of techniques related to process architecture [4–7] that explicate the relationships between the processes. However, these techniques have no ways of expressing the internality of the process, and, again, they use a rather complicated diagrammatic way of presentation of the process relationships.

The question arises whether there could be found a way/technique to represent both sides of the process, its position inside the enterprise and its internal features so that a process model built with this technique could be used for decisions related to the process in question. Such a model should present essential internal and external features of the process without overburdening it with details that are needed for implementing the decision, but not for making it. The goal of this idea paper is to suggest one such technique, which we call Business Process Canvas or BPC for short.

The development of BPC has been inspired by Business Model Canvas (BMC) [8], which, in our view represents an enterprise model in a shell. BMC represents both sides of the enterprise, its internalities, e.g. main activities, and the context, e.g. who are the customers and main partners. Different parts of the canvas need to be aligned with each other, for example, a value proposition should be attractive enough to at least one market segment. However, this alignment is not represented in canvas itself, but is assumed by the creator and the reader of a business model. The model is quite informal, and requires general knowledge about the enterprise/organizational world from its reader, but this is exactly what the managers have. The mentioned above features are the main reason of the popularity of BMC in the business community. BMC gives a concise overview of the business that can be used in the decision-making. In this paper, we are applying the same idea as is embedded in BMC to the realm of business processes, i.e. having a business process model in a nutshell, which is not fully formalized, but which highlights the important parts of the process context and its internality.

To check that such a modeling technique had not already been proposed, we searched the Internet on various combination of keywords *business process* and *canvas*. During the search, we did not find any match in the sense suggested above. The nearest we came to was Process Model Canvas (PMC) [9], but this canvas is designed for helping to build a standard process model, rather than serve as a process model in a nutshell itself. We also found other canvases related to business management, e.g. Operating Model Canvas [10], but none of them was aimed to represent a business processes.

A similar to the canvas concept is a business-oriented template. Such templates are used both in academia, e.g. [11] and practice [12]. While the former refers to a work system, which is more general than a business process, the latter concerns business processes. Usually, such a template has a form of a table that can be quite long, 31 pages in [12]. The absence of the layout in the table-based template makes it difficult to see the connections between its various components which is needed for the template could be considered as a process model in a nutshell. So far, we have not found any process model template that could be used for this purpose. Actually, the goals when designing these template are different, being connected to a specific business task, such as business analysis [11], or process design [12].

The rest of the paper is structured in the following way. In Sect. 2, we briefly overview our research approach and knowledge base used in the development of BPC. In Sect. 3, we describe the canvas layout and each of its sections. In Sect. 4, we present an example of a process and a canvas that corresponds to this process. In the concluding Sect. 5, we discuss possible areas of applicability for the BPC.

2 Research Approach

The research presented in this paper belongs to the Design Science (DS) paradigm, [13] which seeks new solutions for problems known or unknown. In particular, we use the DS interpretation suggested in [14]. Using DS is natural for this research, as the objective is to develop a new way of depicting a business process. The problem the research addresses is to have a business process model that depicts both the process context, i.e. a position of the process in the enterprise ecosystem, and important internal characteristics of the process. The requirements on a new way of modeling the process are to have a model in a nutshell that includes important components/characteristics of the process, but is not overburdened with details. The primary usage of the new modeling technique is human decision-making, thus there is no requirement on the model to be formal.

The proposed solution, or artifact in terminology of [13], is a business model canvas – a set of concerns with a specific layout, that allows to write a text in a natural language related to a specific concern. There is a logical connection between the categories, which is explained, but not expressed in a formal way, e.g. with arrows. When designing the structure of the canvas, we used the knowledge base on business processes from the literature, as well as own experience in analyzing, modeling and supporting business processes. The literature sources are referred to in the next section when describing the structure of BPC. Note that due to the limitation on space, we could not refer to all relevant literature.

The major concepts used for designing the canvas are as follows. The canvas utilizes the duality of the concept of business process that encompasses two sub-concepts: (1) business process instance, otherwise called case or run, and (2) business process type. While the first refer to a particular chain of events with a particular goal, like selling a Volvo S70 to Sven Anderson, the second refers to all process instances of the same kind, e.g. selling personal cars to private customers at a given car dealer. Based on this dichotomy, two distinct systems can be identified in relation to the business process:

1. A temporal system that corresponds to a process instance, which is created at the process start, and disbanded when the goal of the instance has been achieved [4]. This is a respondent system in terminology of [15].
2. A permanent socio-technical system that is responsible for starting and monitoring process instances, and supplying them with resources/assets needed for attaining the instances operational goals, such as people, tools, procedures, etc. [16].

The differentiation underpins the layout and structure of the canvas presented in the next section.

3 The Structure of the Business Process Canvas

The canvas layout we designed is presented in Fig. 1. The layout uses colors to differentiate three distinct sections called:

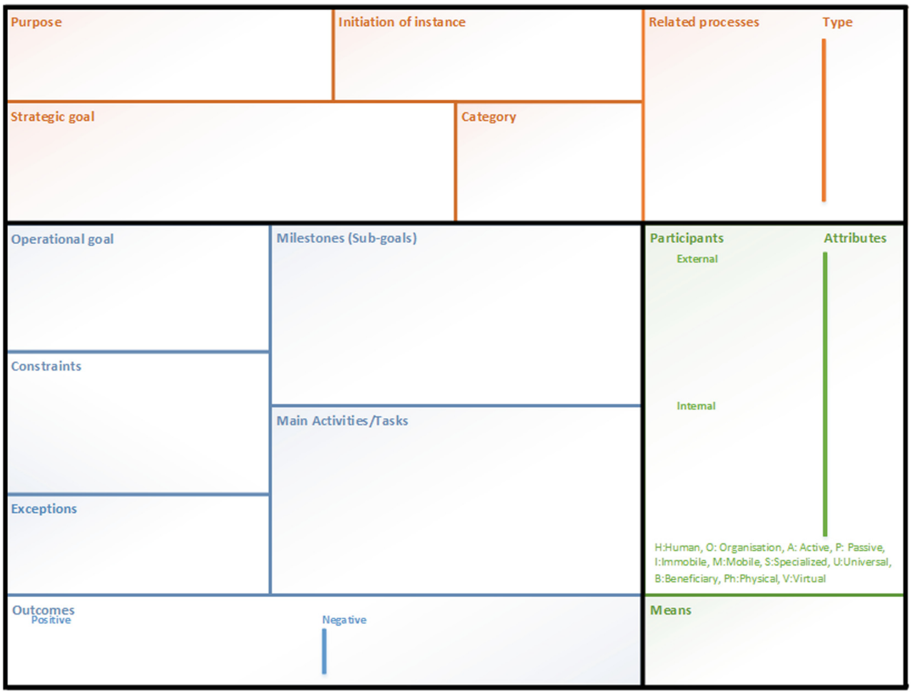


Fig. 1. Canvas outline. (Color figure online)

1. *Outside* – upper part of the canvas marked with the orange color
2. *Inside* – left bottom part of the canvas marked with the blue color
3. *Resources* – right bottom part of the canvas marked with the green color

The distinction between *Outside* and *Inside* is based on the distinction between the process type - *Outside*, and process instance – *Inside*. The outside section of the canvas describes the characteristics and the context of the system related to business process type, e.g. why the process exists, which processes are related to it, etc. In other words, Outside *positions* the process in the ecosystem of the enterprise. The inside section of the canvas describes the characteristics and the context of instance system, e.g. what the operational goal of the individual instance is, which activities/task are completed, etc. In other words, *Inside* presents the details of the instance *operation*. This part can be considered as a generic model of an instance that is valid for any instance of the given type. The third section describes resources engaged in the process instances, such as process participants, tools, specific methods or methodologies, which are provided by the type system to the instance systems.

In the sections below, we describe each of the canvas sections in detail by listing and explaining its components and providing examples of how to fill them.

3.1 Outside Section

The *outside* section describes the characteristics and context of the type system, and includes *Purpose*, *Category*, *Strategic goal*, *Related process*es and *Initiation* of *instances*, that are described below.

Purpose. The primary aim of the process - a general description of why the process exists, without including detailed information, e.g. delivering ordered books to the customer, hiring factory workers, training new employees, providing guarantee and after guarantee service to sold process, etc.

Strategic Goal. A strategic goal adds details to the purpose and lists additional goals that should be achieved by a process if it is properly designed, e.g. ensure getting profit, or using the existing resources, or using partnership with somebody else. It can also include ensuring compliance with current regulations. In addition to the purpose, the strategic goal explains why the process is or should be driven in a particular way [3].

Note that the concept of purpose and strategic goal is a topic much discussed in the literature, though without splitting the concept into two parts. For example it appears in [17] under the name soft goal and in associating to the process type. The soft goal explains why the process exists or should exist in an organization. According to [18], a goal refers to an attainable, measurable and time-bound state of the world that should be achieved or/and sustained. [19] states that the definition of business process goals should initiate the questions "What are we trying to achieve?" and "What are we trying to avoid?". Therefore, defining goals could be done as statements of what the business process is aimed to achieve or avoid. Goals may be qualitative or quantitative. Quantitative goals are easily measurable through some value that should be achieved, while qualitative goals require human judgement to verify whether the goal has been achieved [20]. However, in both cases, there are some important attributes that need to be included when stating a process goal which have been summarized by [18]. These are: attainability, measurability and time-boundedness. Strategic goals tend to be long term goals and are usually stated qualitatively [18].

Category. A classification of the process in one of three types: *Main*, *Supporting*, *Strategic*. The classification of processes in main and supporting can be found in many works. Here we follow the definitions presented in [4], i.e.:

- *Main* – the process that delivers value to the customer/beneficiary, e.g. providing a service.
- *Supporting*, the process facilitating or enable the execution of other business processes. In [4] supporting process is aimed at managing assets/resources used in other processes, e.g. hiring or training the company staff.

To this two we added a category of *strategic* processes aimed at introducing a radical change that can affect the whole organization, including creating new business processes or discontinuing the old ones [21].

Note that a process can belong to more than one category. For example, a service for sold products provides value to the customer (Main process), but it can also be viewed as a supporting process for retaining the customers (which is an asset of the main process [4]). The category is tightly connected to the *purpose* and *strategic goal* of the process.

Initiation of Instances. This component describes the way(s) of how a process instance can be initiated. The description can be a reference to an externally initiated event, e.g. a customer call, or to a situation detected by another process, e.g. a customer prospecting process identifying a potential client against which a sales process instance is initiated.

In a general way, initiation of instances could be defined as a set of conditions that identifies a situation that triggers creating a respondent system – process instance. In a simple case, detecting the triggering situation could be a simple matter: the customer calls and expresses an interest in the company's product or services. In more complicated cases, a monitoring process is required to detect a triggering situation, e.g. customer prospecting, see the discussion on this issue in [21]. Despite the differences, all ways of triggering can be presented as conditions.

Note that *initiation of instances* is not the only component that can be described as a set of conditions. Other components, such as operational goal, milestones, constraints, exceptions, and outcome could also be described in form of conditions (see Sect. 3.2).

Related Processes. The processes that exist in the organization and that are directly or indirectly connected to the given one, together with their types. The *type* of related processes has the same meaning as *category* discussed earlier, another label being used to avoid confusion when reading the canvas.

Listing related processes is important for process change management [17]. Introducing changes in the given process may require changing related processes so that the whole process ecosystem remains in balance.

The Fractal Enterprise Model [4] could be used for identifying, at least part of, related processes. Supporting processes that manage assets for the given process can be listed as related processes. For example, the sales process that delivers new customers (an asset) to the service process can be listed as a related process to a service process. One can also go in the opposite direction listing as related a process which asset the given supporting process manages. For example, the process of hiring workers (an asset) for the factory floor could have the factory production process listed among its related processes.

3.2 Inside Section

The *inside* section describes the characteristics and operation of the instance (respondent) system, and includes *Operational goal*, *Milestones*, *Main activities/Tasks, Constraints*, *Exceptions* and *Outcomes*, that are described below.

Operational Goal. An operational goal defines what an instance of the given process type is aimed to achieve. It can be expressed as a condition on when the instance could be considered as finished and the instance system can be disbanded [3]. For example, for the order handling process the goal can be defined as the ordered goods delivered to the customer and a sum of money agreed upon is received. The conditions should include all possible outcomes (legitimate ends) of a process instance, after it starts, including the customer returning the goods and the company returning the money. Some of these ends are considered positive, others - negative, which are discussed in the text under the *Outcomes*.

Another term used in the literature for *operational goal* is *hard goal* [17] being associated with the process instance. Achieved operational goals, especially with positive outcomes, can be considered as short-term contributions to strategic goals. In addition, they provide a means to measure the progress towards the achievement of a strategic goal [18].

Milestones (Sub-goals). A process instance starts with the conditions of initiation, and ends with the conditions defined by the operational goal. As the transformation cannot be done in one go, there are a number of intermediate process instance states that could be controlled in the frame of the process instance. These are often called *milestones*, and they represent sub-goals to be achieved on the way to achieving the operational goal of the given instance. For example, for an order handling process, such milestones could be *order received*, *goods delivered*, *money received*.

Note that other works, e.g. [22], treat goals in connection to activities, i.e. as attributes of activities. The activities are decomposed to sub-activities, tasks into sub-tasks and goals into sub-goals. We consider the milestones/sub-goals as primary elements of the process description, while activities are considered as the way of attaining the goal or one od sub-goals. This does not create any contradiction, as [22] allows defining several activities for reaching the same (sub-)goal.

Main Activities/Tasks. This component lists the most important actions performed by participants during the execution of a process instance. For an order handling process, these could be taking order, modifying order, packing, invoicing etc.

In the mainstream business process literature, activities/tasks are considered the main components of the process definition. For example, according to [23] a business process consists of parts with a well-defined order which are known as activities. They can be atomic or composite and they can be performed by one or more participants. [24] defines business process activity as a series of units of work or actions that are included in a process or a series of tasks. A task is defined as a special case of activity that involves a single unit of work or action. In our view, activities/tasks are just means of attaining sub-goals, thus only main activities are listed in the canvas.

Constraints. A set of conditions that impose constraints on how the process instance execution is driven. The condition can be expressed in terms of milestones, activities or both. For example, no product delivery before payment is received establishes the order of attaining the sub-goals, i.e. before starting activities aimed at the milestone *goods delivered*, the milestone *money received*.

Constraints in relation to business processes are much discussed in the business process literature, often under the name business rules. Business rules describe policies, procedures and limitations that concern an organization aiming to achieve its business goals and objectives [25]. They are intended to assert business structure, control or influence the behavior of the business [26]. Business rules can be defined formally and informally. In the context of the canvas, only informal definitions are used.

Exceptions. The term refers to an abnormal situation in the instance development. It can be an unexpected event, like the customer calling and complaining that goods have not arrived, or absence of an event, like money not arriving in time on the company's account, or an activity that should be executed by a certain deadline has not been executed without any information on the delay and its cause.

Listing the most common exceptions in the canvas aims at raising awareness of the needs that they should be handled.

Outcomes. This is a way of classifying legitimate endings of process instances as positive and negative outcomes. For example, delivering goods to the customer and getting payed with a profit margin without exceptions and delay can be considered as a positive outcome. The customer canceling the order, or returning goods could be considered as a negative outcome.

Classifying the outcomes as negative and positive is quite known in the literature, see for example, definitions from [24]:

- Positive outcomes, when the result of the process indicates that the process instance execution successfully delivers value to one or more beneficiaries.
- Negative outcomes, when the completed process instance execution has failed to deliver value to any beneficiary or has only delivered partial value.

3.3 Resources Section

The *resources* section describes physical and virtual objects (things) that are engaged in the process instances. It includes two components *Participants* and *Means*, which are described below.

Participants. Participants are entities that perform activities and tasks during a process instance execution or/and are directly affected, e.g. being transformed by these activities and tasks. Participants that perform actions are called active participants (actors), while participants that are affected by actions are called passive participants in terminology of [27]. A participant can be both active and passive, for example, a patient in the hospital can be considered as both passive and active participant in the process of medical treatment, while, a doctor is purely active participant in this process. The same could be said about a student taught in a school.

In many cases, participants are not identified by a reference to a specific entity, but rather by a reference to a class (role), e.g. a doctor, a patient a student, a teacher etc. Furthermore, a participant may be an individual, e.g. a human or a robot, or of an aggregate type, e.g. an entire organization or an organizational unit. An organizational unit is a set of individuals and systems grouped through a set of characteristics or units that share the same kind of functionality in an organization [23]. Considering organizational units as active participants actors is quite common in the literature, for example, [17] define an actor as an entity or organizational unit that is involved in the execution and realization of a business process.

Beside categorization active/passive, participants can be categorized as internal or external participants based on whether they are based within the organization or organizational unit or part of the process's external context. Examples of internal participants include employees, managers and CEOs. Examples of external participants include suppliers, investors or customers [17]. Customers also belong to a special case of participants that receive the value produced by the execution of a process instance. This special type of participant is called beneficiary in [4].

Active participants can be further categorized based on their level of specialization, see [27], as:

- totally universal, when they have no specialization in any of the process activities or tasks
- totally specialized when they have absolute specialization in the performance of certain process activities or tasks
- any state in between

Both active and passive participants can be characterized by their degree of mobility, see [27], as:

- totally immobile, when they cannot be moved to another location
- totally mobile, when they can be moved without substantial costs
- any state in between

Passive participants can be further categorized based on their status of physicalness, see [27], as:

- physical, when they are material and tangible
- virtual, when they are immaterial and intangible

In the *participants* component of the canvas, main participants are listed along with initial letters that correspond to their attributes discussed above. A legend at the bottom of this canvas component includes all the attributes that are deemed essential for decision making related to a particular business process.

Means. Means are tools, techniques and methods used by process instance participants to perform activities and tasks during the execution of an instance. Means corresponds to the control elements of a respondent system [15], see also [4]. Typical examples are email, eXcel, ERP, CRM, SCRUM, Lean, Three sigma etc.

In the terminology of fractal enterprise model [4], tools include assets of the infrastructure type and executable templates, e.g. process maps, operational manuals, etc. In a similar way to categorization of passive participants, tools can also be categorized according to their mobility and physicalness. Examples of physical tools include factory equipment, printed documents and vehicles, while examples of virtual tools include software and manuals in digital documents. The canvas component associated with tools does not require adding attributes letters to the tools names, though the tool name and its short description could indicate which attributes they have.

4 An Example

In this section, we demonstrate how to fill the canvas using a process from a business case used for teaching state-oriented business process modeling [28]. Information about the process itself and its context is available in forms of interviews with the stakeholders of the company that intends to run the process. The interviews in a voice recorded form, and other background information is available from [29]. Both the interviews and background information are real, except that the name of the company has been changed.

Harmony Inside AB is a Swedish company that provides services and products to the individuals who suffers from Irritable Bowel Syndrome (IBS) using the FODMAP method. The company is a startup coming into the growth phase. It was started by two dietitians that are now expanding their business and moving it to the web. The products and services include cooking recipes, food store, books, general advice, an online course and individual consultations.

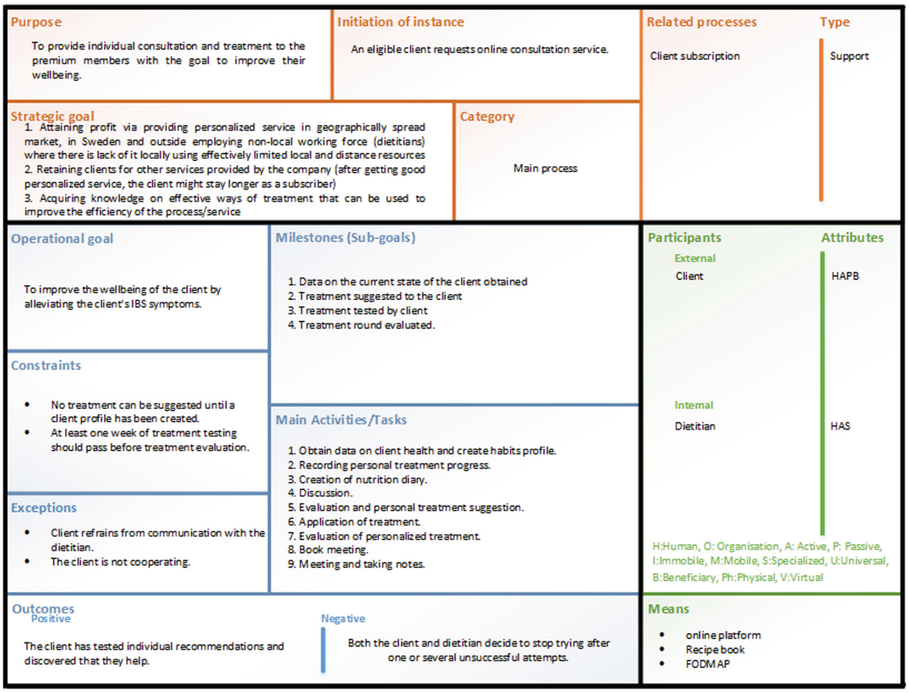
The chosen method of payment is membership with monthly fees. There are three levels of member subscription: Starter, Standard and Premium. The first two levels concern one way only interaction, cooking recipes, general advice, an online course and some other services. The third level of membership includes individual consultation and treatment by a dietitian through an online platform. The process of providing such consultation and treatment is used as an example in this paper. The canvas for this process is presented in Fig. 2, and its components are listed and explained below. Note that we did not try to fully fill all components, only representative examples for each component are presented in the canvas.

Purpose. The purpose of the process is to provide individual consultation and treatment to the premium members with the goal to improve their wellbeing.

Strategic Goal. There are more than one strategic goals associated to the online consultation process. First of all, the company aims to attain profit in a geographically widespread market, not only in Sweden but also in other countries, by using effectively local and distance resources to provide personalized services. Another goal is to retain its clients as subscribers for other services provided by the company by providing personalized services of the highest quality. Finally, the company aims to acquire more knowledge on effective ways of treatment that can be used in order to improve the efficiency of the currently available treatment.

Category. The online consultation business process can be classified as a main process since it exists to provide value to its beneficiaries.

Initiation of Instance. The process instance execution is initiated when an eligible customer requests to participate in the online consultation service.



Purpose

To provide individual consultation and treatment to the premium members with the goal to improve their wellbeing.

Initiation of instance

An eligible client requests online consultation service.

Related processes

Client subscription

Type

Support

Strategic goal

1. Attaining profit via providing personalized service in geographically spread market, in Sweden and outside employing non-local working force (dietitians) where there is lack of it locally using effectively limited local and distance resources
2. Retaining clients for other services provided by the company (after getting good personalized service, the client might stay longer as a subscriber)
3. Acquiring knowledge on effective ways of treatment that can be used to improve the efficiency of the process/service

Category

Main process

Operational goal

To improve the wellbeing of the client by alleviating the client's IBS symptoms.

Milestones (Sub-goals)

1. Data on the current state of the client obtained
2. Treatment suggested to the client
3. Treatment tested by client
4. Treatment round evaluated.

Participants

External

Client

Internal

Dietitian

Attributes

HAPB

HAS

H:Human, O: Organisation, A: Active, P: Passive, I:Immobile, M:Mobile, S:Specialized, U:Universal, B:Beneficiary, Ph:Physical, V:Virtual

Constraints

- No treatment can be suggested until a client profile has been created.
- At least one week of treatment testing should pass before treatment evaluation.

Main Activities/Tasks

1. Obtain data on client health and create habits profile.
2. Recording personal treatment progress.
3. Creation of nutrition diary.
4. Discussion.
5. Evaluation and personal treatment suggestion.
6. Application of treatment.
7. Evaluation of personalized treatment.
8. Book meeting.
9. Meeting and taking notes.

Exceptions

- Client refrains from communication with the dietitian.
- The client is not cooperating.

Outcomes

Positive

The client has tested individual recommendations and discovered that they help.

Negative

Both the client and dietitian decide to stop trying after one or several unsuccessful attempts.

Means

- online platform
- Recipe book
- FODMAP

Fig. 2. The Harmony Inside online consulting business process in a nutshell using the BPC.

Related Processes. One process that is related to the online consultation is the client subscription process. More related processes exist, but we limit ourselves to one in this example.

Operational Goal. The operational goal of a process instance is to improve the wellbeing of the client by alleviating the client's IBS symptoms. The instance ends when the client has tested individual recommendations and discovered that they help, or both the client and dietitian decide to stop trying after one or several unsuccessful attempts. The first case represents a positive outcome, while the second – a negative one.

Milestones. The following milestones are identified for this process: (i) data on the current state of the client obtained, (ii) treatment suggested to the client, (iii) treatment tested by client, and (iv) treatment round evaluated.

Main Activities & Tasks. The main activities and tasks that are performed during a process instance execution are (i) obtain data on client health and create habits profile, (ii) recording personal treatment progress, (iii) creation of nutrition diary, (iv) discussion, (v) evaluation and personal treatment suggestion, (vi) application of treatment, (vii) evaluation of personal treatment, (viii) book meeting and (ix) meeting and taking notes.

Constraints. Examples of the constraints for this process include:

- No treatment can be suggested until a client profile has been created.
- At least one week of treatment testing should pass before treatment evaluation.

Exceptions. For this process, two typical exceptions are presented in the canvas: (1) the client unexpectedly refrains from communication with the dietitian, (2) the client is not cooperating, e.g. not following the treatment recommendations.

Outcomes. See two outcomes mentioned in the operational goal.

Participants. There are only two participants in this process. The dietitian, who bears the attributes, Human, Active and Specialized and the client who bears the attributes Human, Active, Passive and Beneficiary. Both participants are performing actions in an instance; however, the client is also being changed by these actions, therefore, bears, both the active and passive attributes. In addition, the client is the one that receives value from the process, being a beneficiary.

Means. The means that are used during online consultation are the online platform that enables interaction, the recipe books, used to provide recipes, and the FODMAP method.

5 Conclusion - Areas of Applicability and Future Research

5.1 Contribution

The main contribution of this paper is developing an idea of having a business process canvas as a business process model in a nutshell. To the best of our knowledge, such an idea has not been suggested before. The nearest suggestions in the form of business model canvas [8], and operating model canvas [10] were not designed having the "nutshell" purpose in mind.

When developing the nutshell idea, we decided on sections - *Outside*, *Inside* and *Resources* - and their content, as well as the canvas layout, which is presented in Fig. 1. The development of the idea is in its initial phase; thus, we do not insist that the canvas presented in Fig. 1 is the final one. Though we believe that the sections we identified are the right ones, the components of each section and the canvas layout are subjects to further development and revision. What we wanted to demonstrate in this paper is that such a canvas can be rationally built.

5.2 Possible Areas of Application

As has been discussed in the introduction, the main motivation of creating a BPC is to give the management a model of business process that they understand and can use in decision-making. Thus, the first area of application of BPC is **decision making**. For this end, the canvas should be fully built, so that the decision related to the given process could be debated and made. The main idea is that all parts of the canvas should be in balance. The canvas can be used for finding a cause of problems in the process and for planning changes.

Finding a cause of a problem consists of investigating whether all parts of the canvas correspond to each other. For example, if a strategic goal lists “being flexible in relation to the customer needs” and the constraint component sets many constraints on the instances’ behavior, there clearly exists a contradiction that should be removed. Either constraints should be revised, or flexibility should be removed from the goals.

Planning the change is done by introducing changes in one or several components, and then removing the contradictions that appear by adjusting other components. The latter may concern introducing changes in related processes.

The second area, where BPC can be used is a preliminary phase for **process modeling**. While investigating the process of creating process models, [30] identified that prior structuring of domain knowledge significantly affected in a positive way both the outcome and the process of process modeling, when casual process modelers are concerned. In addition, certain casual process modeling professional roles choose a faster method that employs graphical approaches and the use of post-its and textual notes, in other words, the Canvas approach [31]. The suggested canvas, along with any similar canvas approach, not only depicts a process in a nutshell, but also provides a process modeler with a structuring of domain knowledge that bridges this gap. The process of modeling can be considered as a process of enrichment or elaboration where the modeler starts with very simple and abstract models, and moving in an evolutionary fashion models with greater details depicting the complexity of the case being modeled [32]. The suggested canvas can play the role of an initial step in this process.

Note that for the modeling purposes BPC has competitors, e.g. business model canvas [8]. The distinctive feature of BPC in respect to modeling, however, is that BPC has no connection to a particular modeling technique. Thus, it could be used for any kind of modeling, be it workflow modeling, state-oriented modeling, goal-oriented modeling, etc.

When using BPC for modeling purposes, the direction of filling the canvas can be chosen as “from inside out”. This means that first one fills the components of the *inside* section, then goes to *resources*, and to the *outside* section.

The third area of possible BPC application is new **process design**. In this case, one uses the direction “from outside in”; filling the canvas starts with defining purpose, strategic goal, and other components of the outside section. After that, resources that are needed are identified and operational details of the process instances designed.

5.3 Directions for Future Research

The next stage of canvas development is testing it in practice, which will give feedback of what is needed for further development. The testing needs to be done for all three potential areas of application: decision-making, modeling and design. Though BPC is being designed primarily for the first and third areas of application, finding a case of these kinds might require more efforts than testing BPC for modeling. Therefore, testing BPC for modeling is planned as the first step in further development of the idea. To start with, the canvas will be introduced to the students studying Business Process and Case management [28] and completing their Master theses in this field.

References

1. Hoverstadt, P.: Why business should take enterprise architecture seriously. In: Gøtze, J., Jensen-Waud, A. (eds.) Beyond Alignment: Applying Systems Thinking in Architecting Enterprises, pp. 55–166. College Publications, London (2013)
2. van der Aalst, W.M.P., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT Press, Cambridge (2002)
3. Bider, I.: State-oriented business process modeling: principles, theory and practice (2002)
4. Bider, I., Perjons, E., Elias, M., Johannesson, P.: A fractal enterprise model and its application for business development. Softw. Syst. Model. **16**, 663–689 (2016)
5. Dijkman, R., Vanderfeesten, I., Reijers, H.A.: Business process architectures: overview, comparison and framework. Enterp. Inf. Syst. **10**, 129–158 (2016)
6. Green, S., Ould, M.: A framework for classifying and evaluating process architecture methods. Softw. Process Improv. Pract. **10**, 415–425 (2005)
7. Koliadis, G., Ghose, A.K., Padmanabhuni, S.: Towards an enterprise business process architecture standard. In: 2008 IEEE Congress on Services - Part I, Honolulu, HI, USA, pp. 239–246. IEEE (2008)
8. Osterwalder, A., Pigneur, Y., Clark, T.: Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. Wiley, Hoboken (2010)
9. Bijl, M., Devlin, J.P., Ruting, D.: Process Model Canvas. http://www.processmodelcanvas.com/
10. Campbell, A., Gutierrez, M., Lancelott, M.: Operating Model Canvas: Aligning Operations and Organization with Strategy. Van Haren Publishing, Zaltbommel (2017)
11. Alter, S.: Work system theory: overview of core concepts, extensions, and challenges for the future. JAIS **14**, 72–121 (2013)
12. Business Process Design Templates. http://klariti.com/business-process-design-template/
13. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**, 75–105 (2004)
14. Bider, I., Johannesson, P., Perjons, E.: Design science research as movement between individual and generic situation-problem–solution spaces. In: Baskerville, R., De Marco, M., Spagnoletti, P. (eds.) Designing Organizational Systems, pp. 35–61. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33371-2_3
15. Lawson, H.W.: A Journey Through the Systems Landscape. College Publications, London (2010)
16. Bider, I., Kowalski, S.: A framework for synchronizing human behavior, processes and support systems using a socio-technical approach. In: Bider, I., Gaaloul, K., Krogstie, J., Nurcan, S., Proper, H.A., Schmidt, R., Soffer, P. (eds.) BPMDS/EMMSAD 2014. LNBIP, vol. 175, pp. 109–123. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43745-2_8
17. Elias, M., Shahzad, K., Johannesson, P.: A business process metadata model for a process model repository. In: Bider, I., Halpin, T., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Ukor, R. (eds.) BPMDS/EMMSAD 2010. LNBIP, vol. 50, pp. 287–300. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13051-9_24
18. Markovic, I., Kowalkiewicz, M.: Linking business goals to process models in semantic business process modeling. In: Enterprise Distributed Object Computing Conference, Munich, Germany, pp. 332–338. IEEE (2008)
19. Kueng, P., Kawalek, P.: Goal-based business process models: creation and evaluation. Bus. Process Manag. J. **3**, 17–38 (1997)

20. Vasconcelos, A., Caetano, A., Neves, J., Sinogas, P., Mendes, R., Tribolet, J.: A framework for modeling strategy, business processes and information systems. In: Proceedings Fifth IEEE International Enterprise Distributed Object Computing Conference, Seattle, WA, USA, pp. 69–80. IEEE Computer Society (2001)
21. Bider, I., Bellinger, G., Perjons, E.: Modeling an agile enterprise: reconciling systems and process thinking. In: Johannesson, P., Krogstie, J., Opdahl, A.L. (eds.) PoEM 2011. LNBIP, vol. 92, pp. 238–252. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24849-8_18
22. Yu, E.S.K., Mylopoulos, J.: Using goals, rules, and methods to support reasoning in business process reengineering. In: 1994 Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences, Wailea, HI, USA, pp. 234–243. IEEE Computer Society Press (1994)
23. Ouali, S., Mhiri, M., Bouzguenda, L.: A multidimensional knowledge model for business process modeling. Procedia Comput. Sci. **96**, 654–663 (2016)
24. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
25. Salgado, C.E., Teixeira, J., Machado, R.J., Maciel, R.S.P.: Generating a business model canvas through elicitation of business goals and rules from process-level use cases. In: Johansson, B., Andersson, B., Holmberg, N. (eds.) BIR 2014. LNBIP, vol. 194, pp. 276–289. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11370-8_20
26. Eijndhoven, T. van, Iacob, M.-E., Ponisio, M.L.: Achieving business process flexibility with business rules. In: Enterprise Distributed Object Computing Conference, Munich, Germany, pp. 95–104. IEEE (2008)
27. Bider, I., Perjons, E.: Evaluating adequacy of business process modeling approaches. In: Wang, M., Sun, Z. (eds.) Handbook of Research on Complex Dynamic Process Management: Techniques for Adaptability in Turbulent Environments, pp. 79–102. IGI Global, Hershey (2010)
28. Koutsopoulos, G., Bider, I.: Teaching and learning state-oriented business process modeling. experience report. In: Reinhartz-Berger, I., Gulden, J., Nurcan, S., Guédria, W., Bera, P. (eds.) BPMDS/EMMSAD 2017. LNBIP, vol. 287, pp. 171–185. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59466-8_11
29. Harmony Online Consultation Project – A ST Transformer Project for Harmony Inside. http://harmonyproject.blogs.dsv.su.se/
30. Pinggera, J., Zugal, S., Weber, B., Fahland, D., Weidlich, M., Mendling, J., Reijers, H.A.: How the structuring of domain knowledge helps casual process modelers. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) ER 2010. LNCS, vol. 6412, pp. 445–451. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16373-9_33
31. Glória Júnior, I., Gonçalves, R.F.: The identification of the professional profile that uses canvas approach. In: Nääs, I., Vendrametto, O., Reis, J.M., Gonçalves, R.F., Silva, M.T., von Cieminski, G., Kiritsis, D. (eds.) APMS 2016. IAICT, vol. 488, pp. 544–551. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-51133-7_65
32. Morris, W.T.: On the art of modeling. Manag. Sci. **13**, B707–B717 (1967)

Automatic Analysis of Business Processes (BPMDS 2018)

Identifying Candidate Tasks for Robotic Process Automation in Textual Process Descriptions

Henrik Leopold, Han van der Aa(✉), and Hajo A. Reijers

Vrije Universiteit Amsterdam, De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
{h.leopold,j.h.vander.aa,h.a.reijers}@vu.nl

Abstract. The continuous digitization requires organizations to improve the automation of their business processes. Among others, this has lead to an increased interest in Robotic Process Automation (RPA). RPA solutions emerge in the form of software that automatically executes repetitive and routine tasks. While the benefits of RPA on cost savings and other relevant performance indicators have been demonstrated in different contexts, one of the key challenges for RPA endeavors is to effectively identify processes and tasks that are suitable for automation. Textual process descriptions, such as work instructions, provide rich and important insights about this matter. However, organizations often maintain hundreds or even thousands of them, which makes a manual analysis unfeasible for larger organizations. Recognizing the large manual effort required to determine the current degree of automation in an organization's business processes, we use this paper to propose an approach that is able to automatically do so. More specifically, we leverage supervised machine learning to automatically identify whether a task described in a textual process description is manual, an interaction of a human with an information system or automated. An evaluation with a set of 424 activities from a total of 47 textual process descriptions demonstrates that our approach produces satisfactory results.

1 Introduction

Many organizations currently face the challenge of keeping up with the increasing digitization. Among others, it requires them to adapt existing business models and to respectively improve the automation of their business processes [28]. While the former is a rather strategic task, the latter calls for specific operational solutions. One of the most recent developments to increase the level of automation is referred to as Robotic Process Automation (RPA). In essence, RPA emerges in the form of software-based solutions that automatically execute repetitive and routine tasks [5]. In this way, knowledge workers can dedicate their time and effort to more complex and value adding tasks.

While the benefits of RPA have been demonstrated in different contexts [7,20], one of the key challenges is to effectively identify processes and tasks that are suitable for automation [5]. So far, research has focused on the establishment of criteria [11,37] and step-by-step guidelines [9] as means to support organizations in addressing this challenge. However, what all these methods have in common is that they require a manual analysis of the current *degree of automation*, i.e., they depend on the manual identification of tasks and (sub-)processes that are automated or supported by an information system. This identification task requires a thorough analysis of process-related documentations such as process models and textual process documentations. While especially the latter often provides rich and detailed insights, organizations typically maintain hundreds or even thousands of them [3]. As a result, these methods do not scale for organizations with hundreds of processes

Recognizing the large manual effort required to determine the current degree of automation in an organization's business processes, we use this paper to propose an approach that is able to automatically do so. More specifically, we combine supervised machine learning and natural language processing techniques to automatically identify whether a task described in a textual process description is a (1) manual task, (2) user task (interaction of a human with an information system) or (3) automated task. An evaluation with a set of 424 activities from a total 47 textual process descriptions demonstrates that our approach produces satisfactory results. Therefore, our approach can be employed to reduce the effort required to determine the degree of automation in an organization's processes, as a first step in RPA endeavors.

The rest of the paper is organized as follows. Section 2 illustrates the problem we address using a running example. Section 3 introduces our approach for automatically determining the degree of automation of textual process descriptions on a conceptual level. Section 4 presents the results of our evaluation. Section 5 discusses related work before Sect. 6 concludes the paper.

2 Problem Statement

In this section, we illustrate the problem of automatically identifying the degree of automation of tasks described in a textual process description. Building on the three categories of task automation introduced in [9], our goal is to classify each task from a given textual process description as either (1) manual, (2) user task (interaction of a human with an information system) or (3) automated. Figure 1 shows an exemplary textual process description, the associated relevant process tasks, and their degree of automation.

Figure 1 shows that this textual process description contains two manual tasks, two user tasks, and two automated tasks. The manual tasks include the decision of the supervisor about the vacation request (task 3) and the completion of the management procedures by the HR representative (task 6). The user tasks are the two tasks in the process that are executed using the help of an information system. That is, the submission and the reception of the vacation request (tasks 1 and 2). The automated tasks are tasks executed by the ERP system. This includes returning the application to the employee (task 4) as well as generating

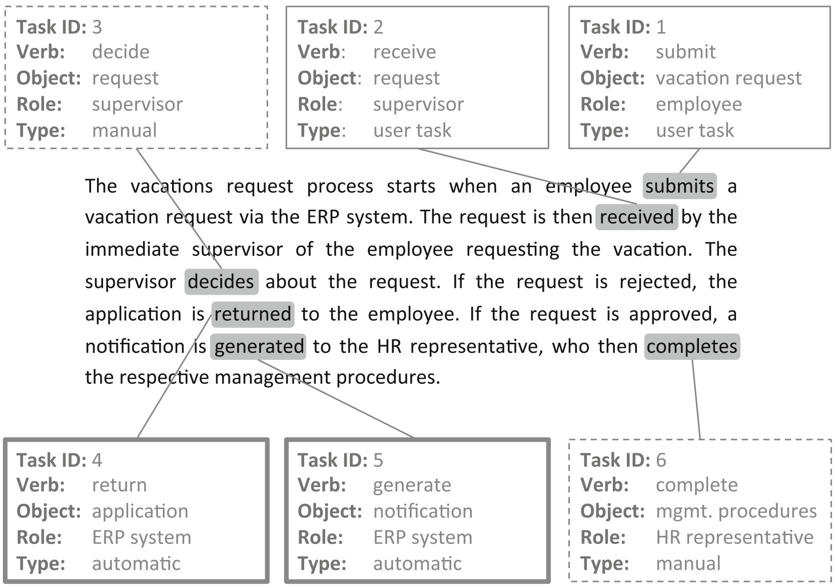

Task ID: 3
Verb: decide
Object: request
Role: supervisor
Type: manual
Task ID: 2
Verb: receive
Object: request
Role: supervisor
Type: user task
Task ID: 1
Verb: submit
Object: vacation request
Role: employee
Type: user task
The vacations request process starts when an employee submits a vacation request via the ERP system. The request is then received by the immediate supervisor of the employee requesting the vacation. The supervisor decides about the request. If the request is rejected, the application is returned to the employee. If the request is approved, a notification is generated to the HR representative, who then completes the respective management procedures.
Task ID: 4
Verb: return
Object: application
Role: ERP system
Type: automatic
Task ID: 5
Verb: generate
Object: notification
Role: ERP system
Type: automatic
Task ID: 6
Verb: complete
Object: mgmt. procedures
Role: HR representative
Type: manual

Fig. 1. Process description with highlighted activities and their degree of automation

the notification to the HR representative (task 5). Analyzing this scenario in more detail, reveals that the automatic classification of these tasks is associated with two main challenges:

1. *Identification of tasks*: Before a task can be classified, an automated approach must be able to detect the tasks described in a text. Note, for example, that the verb "*starts*", the verb "*rejected*" as well as the verb "*approved*" do not relate to tasks. The first is not relevant to the classification task at hand because it represents a piece of meta information about the process. The latter two tasks are not relevant because they rather relate to conditions than to tasks, i.e., "*if the request is rejected*" describes a state, rather than an activity being performed. Besides identifying relevant verbs, the identification of tasks also requires to properly infer the *object* to which a verb refers and the resource that executes the task, i.e. the *role*.
2. *Consideration of context*: To reliably predict whether a certain activity is a manual, user, or automated task, an automated approach must be able to take a number of contextual factors into account. Consider, for instance, the receipt of the vacation request (task 2). While in this process description the request is submitted to an information system, this might not be the case in other processes (a request could be also received orally or in writing). The fact that an information system is mentioned in the first sentence, must respectively be considered when classifying a task described later in the process.

In prior work, only the former challenge has been addressed. The technique for generating process models from natural language texts proposed by

Friedrich et al. [10] can reliably recognize and extract tasks from textual process descriptions. To the best of our knowledge, there does not exist any technique that addresses the second challenge. In this paper, we do so by operationalizing the problem of automatically identifying the degree of task automation as multi-class classification problem. In the next section, we elaborate on the details of our proposed solution.

3 Conceptual Approach

In this section, we present our approach for automatically identifying the degree of automation of tasks described in a textual process description. Section 3.1 first gives an overview of the approach. Section 3.2 introduces the dataset we use in this paper, before Sect. 3.3 through Sect. 3.5 elaborate on the details of our approach.

3.1 Overview

The overall architecture of our three-step approach is visualized in Fig. 2. The approach takes as input a textual process descriptions and returns a list of process tasks that are classified according to their degree of automation.

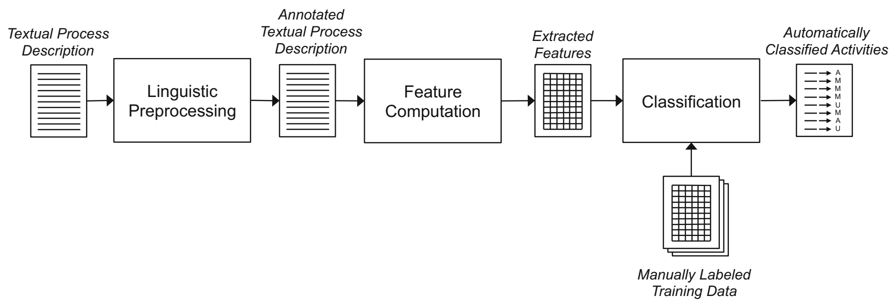

Textual Process Description
Linguistic Preprocessing
Annotated Textual Process Description
Feature Computation
Extracted Features
Classification
Automatically Classified Activities
A
M
M
M
U
M
A
U
Manually Labeled Training Data

Fig. 2. Overview of the proposed approach

The first step is to parse the text and to identify the relevant linguistic entities and relations that denote tasks in a process description. For instance, we determine which words represent verbs and to which objects they relate. The result of this preprocessing step is a textual process description annotated with the linguistic information related to the process' tasks. The second step is the computation of the features we use for prediction. In particular, we compute features related to the verbs and objects that characterize tasks in a process, the resources that execute tasks, and a feature characterizing terms from IT domains. The output of this step is a feature table that contains the extracted tasks and their corresponding features. In the third step, we perform a classification based on the computed features. In the context of this paper, we use an SVM, which is a supervised machine learning algorithm that automatically classifies the input based on a set of manually labeled training instances. The output of the classification is a list of tasks, each automatically classified as manual, user, or automated.

In the following sections, we elaborate on each step in more detail. Because of the supervised nature of our classification approach, we begin with introducing our dataset.

3.2 Dataset

For this paper we use a subset of a collection of textual process descriptions introduced in [10]. The collection contains 47 process descriptions from 10 different industrial and scholarly sources. We removed one of these sources (i.e. 14 process descriptions) because the textual descriptions from this source were obtained using Google Translate and their language quality was insufficient for our purposes. To obtain the required classifications for the 424 tasks described in this dataset, two researchers independently classified each task as manual, user, or automated. Conflicts were resolved by involving a third researcher. Table 1 gives an overview of the characteristics of the resulting dataset.

Table 1. Characteristics of dataset

ID	Source	Type	D	S	W/S	MT	UT	AT
1	HU Berlin	Academic	4	10.0	18.1	52	4	1
2	TU Berlin	Academic	2	34.0	21.2	42	38	11
3	QUT	Academic	8	6.1	18.3	51	20	1
4	TU Eindhoven	Academic	1	40.0	18.5	36	8	0
5	Vendor tutorials	Industry	4	9.0	18.2	9	23	2
6	inubit AG	Industry	4	11.5	18.4	9	23	3
7	BPM Practitioners	Industry	1	7	9.7	7	1	0
8	BPMN practice Handbook	Textbook	3	4.7	17.0	14	6	1
9	BPMN guide	Textbook	6	7.0	20.8	30	30	2
Total			**33**	**9.7**	**16.8**	**250**	**153**	**21**

Legend: D = Number of process descriptions per source, S = Average number of sentences, W/S = Average number of words per sentence, MT = Total number of manual tasks per source, UT = Total number of user tasks per source, AT = Total number of automated tasks per source

The data from Table 1 illustrates that the process descriptions from our dataset differ with respect to many dimensions. Most notably, they differ in size. The average number of sentences ranges from 4.7 to 34.0. The longest process description contains a total of 40 sentences. The descriptions also differ in the average length of the sentences. While the descriptions from the *BPM Practitioners* source contain rather short sentences (9.7 words), the process descriptions from the *TU Berlin* source contain relatively long sentences (21.2 words). The process descriptions also differ with respect to the degree of automation. Some sources contain process descriptions mostly covering manual tasks (e.g. the *HU Berlin* source), others contain a quite considerable number of automated tasks (e.g. the *TU Berlin* source). Lastly, the process descriptions differ in terms of how explicitly and unambiguously they describe the process behavior. Among others, this results from the variety of authors that created the textual descriptions.

3.3 Linguistic Preprocessing

The goal of the linguistic preprocessing step is to automatically extract verbs, object, and roles related to tasks described in the input text. To accomplish this, we build on a technique that was originally developed for the extraction of process models from natural language text [10]. This technique, which is regarded as state-of-the-art [31], combines linguistic tools such as the Stanford Parser [18] and VerbNet [33] to, among others, identify verbs, objects, and roles. The advantage of this technique is its high accuracy and its ability to resolve so-called anaphoric references such as "*it*" and "*they*". To illustrate the working principle of the technique, consider the first sentence from the running example in Fig. 1:

"*The vacations request process starts when an employee submits a vacation request via the ERP system.*"

The first step is the application of the Stanford Parser, which automatically detects the part of speech of each word as well as the grammatical relations between them. The result of the part-of-speech tagging looks as follows.

"*The/DT vacations/NNS request/NN process/NN starts/VBZ when/WRB an/DT employee/NN submits/VBZ a/DT vacation/NN request/NN via/IN the/DT ERP/NNP system/NN ./.*"

We can see that the Stanford Parser correctly identifies two verbs "*starts*" and "*submits*" (indicated by the tag "*VBZ*"). The dependency analysis of the Stanford Parser further reveals to which subjects and objects these verbs relate:

nsubj(starts-5, process-4)
nsubj(submits-9, employee-8)
dobj(submits-9, request-12)
compound(request-12, vacation-11)

The verb "*starts*" relates to the subject "*process*" and the verb "*submits*" relates the subject "*employee*" as well as the object "*request*". The Stanford Parser also recognizes that "*vacation request*" is a compound noun (i.e., a noun that consists of several words). Based on the part-of-speech tagging output and the dependency relations, the technique from [10] automatically extracts task records consisting of a verb, an object, and the executing role. It also recognizes that the verb "*start*" in this context represents meta information and not a relevant task. It is respectively not included as a task record. The final set of task records then represents the input to the next step of our approach.

3.4 Feature Computation

The selection and computation of suitable features is the key task when building a machine learning-based solution [8]. Therefore, we manually analyzed which characteristics in our dataset affect the degree of automation of a task. As a result, we selected and implemented four features:

- *Verb feature (categorical)*
- *Object feature (categorical)*
- *Resource type (human/non-human)*
- *IT domain (yes/no)*

In the following paragraphs we elaborate on the definition and rationale of each feature as well as its computation.

Verb Feature. The *verb* feature is a categorical feature and relates to the verb used in the context of a task. The main idea behind this feature is that certain verbs are more likely to be associated with automated tasks than others. As an example, consider the verbs "*generate*" or "*transmit*", which likely relate to automated tasks. The verbs "*analyze*" and "*decide*", by contrast, are more likely to relate to manual tasks. The advantage of introducing a verb feature over using predefined verb classed (such as the Levin verb classes [27]) is that a verb feature does not tie a verb to a specific automation class. The verb "*generate*", for instance, might as well be used in the context of "*generate ideas*" and, thus, refer to a manual task. Such a context-related use can be taken into account when the verb is considered as part of a set of features.

The computation of this feature is straightforward since it is explicitly included in the task record from the linguistic preprocessing step.

Object Feature. The *object* feature is a categorical feature and captures the object that the verb of the task relates to. The rationale behind this feature is, similar to the verb feature, that certain objects are more likely to be associated with automated tasks than others. As an example, consider the two verb-object combinations "*send letter*" and "*send e-mail*". Although both contain the verb "*send*", the object reveals that the former relates to a manual and the latter relates to a user task (sending an e-email certainly requires the interaction with a computer). While the number of objects we may encounter in textual process descriptions is much higher than the number of verbs, including the object as a feature might still help to differentiate different degrees of task automation.

Similar to the verb feature, the computation of this feature is straightforward since it is part of the task record from the linguistic preprocessing step.

Resource Type Feature. The *resource type* feature is a binary feature that characterizes the resource executing a task as either "*human*" or "*non-human*". The reason for encoding the resource as a binary feature instead of a classical categorical feature is the high number of resources that can execute a task. Depending on the domain of the considered process, resources may, among others, relate to specific roles (e.g. "*manager*" or "*accountant*"), departments (e.g., "*HR department*" or "*accounting department*"), and also systems ("*ERP system*" or "*information system*"). Despite this variety, the key characteristic revealing whether a task is likely to be automated is the type of the resource, that is, whether the resource is human or not. Apparently, a human resource can only relate to a manual or user task, while a non-human resource can also execute automated task (especially when the non-human resource represents an IT system).

Unlike the computation of the verb and the object feature, the computation of the resource type feature is not trivial. The task record from the linguistic preprocessing step only contains the actual resource and no indication of the resource type. To determine the resource type, we use the lexical database WordNet [29]. WordNet groups English words into sets of synonyms, so-called synsets. For each of the 117,000 synsets WordNet contains, it provides short definitions, examples, and a number of semantic relations to other synsets. To compute this feature, we leverage the *hypernym* relationship from WordNet. In general, a hypernym is a more generic term for a given word. For instance, the word "*vehicle*" is the hypernym of "*car*" and the word "*bird*" is the hypernym of "*eagle*". Based on this notion of hypernymy and the hierarchical organization of WordNet, we are able to infer for a given resource whether its hypernym is "*physical entity*", "*abstract entity*" or a "*person*". Based on this hypernym information we then can automatically categorize whether a resource is human or non-human.

IT Domain Feature. The *IT domain* feature is a binary feature that reveals whether a task relates to the IT domain or not. The rationale behind this feature is that a task that relates to the IT domain is likely to be a user task or even an automated task. As example, consider the text fragment "*the customer submits a complaint via the complaint management system*". This fragment contains the human actor "*customer*", the verb "*submit*" and the object "*complaint*". Neither of these elements clearly indicates a degree of automation. However, the fragment also mentions a "*complaint management system*". The goal of the IT domain feature is to take such IT-related context into account.

To compute this feature, we leverage the glossary of computer terms developed by the University of Utah[1]. Besides a comprehensive coverage of technical terms, this list also contains verbs and adjectives that are used in an IT context. If a considered sentence, contains one or more terms from this list, the IT domain feature receives the value "*yes*" for any task that is part of this sentence.

[1] http://www.math.utah.edu/~wisnia/glossary.html.

3.5 Classification

In the final step of our approach, the actual classification of tasks from unseen process descriptions takes place. As described in the previous section, there is not a single feature that independently reveals the degree of automation of a given task. It rather depends on the specific context of the task in the process. To be able to still classify unseen tasks, we employ a Support Vector Machine [6], a supervised machine learning algorithm. The advantages of SVMs are, among others, that they can deal well with relatively small datasets, they have a low risk of overfitting, and they scale well. For these reasons SVMs have also been frequently applied in the context of other text classification tasks [16,35].

The core strategy of an SVM is to find a so-called *hyperplane* that best divides a dataset. While in a two-dimensional space a simple line would be sufficient to do so, an SVM maps the data to higher and higher dimensions until a hyperplane can be formed that clearly segregates the data. Since an SVM is a supervised machine learning algorithm, it needs to be trained on a manually labeled dataset.

In the next section, we describe how we implemented the approach outlined in this section and demonstrate the effectiveness of the approach through a quantitative evaluation.

4 Evaluation

This section reports on our evaluation experiments. We first elaborate on the evaluation setup and the implementation of our approach. Then, we provide a detailed discussion of the results.

4.1 Setup

The goal of the evaluation experiments is to demonstrate that the approach introduced in this paper can reliably determine the degree of automation of previously unseen textual process descriptions. To this end, we implemented our approach as a Java prototype. Besides the code from [10], which we use to extract tasks from a textual process descriptions, we build on the machine learning library Weka [15] to implement the SVM, and JWNL [36] for incorporating and accessing the lexical database WordNet.

To evaluate the performance of our approach, we conducted a repeated 10-fold cross validation using our dataset. The idea behind this validation approach is to randomly split the data set into 10 mutually exclusive subsets (so-called folds) of about equal size. The SVM is then trained on 9 of the 10 folds and tested on the remaining (unseen) fold. This process is repeated 10 times such that, in the end, all data has been used for both training and testing. The advantage of this evaluation method is that it does not require to partition the data set into training and test data. We ran four different configurations of our approach:

1. Training on action feature only (A)
2. Training on action and object feature (A+O)

3. Training on action, object, and resource type feature (A+O+RT)
4. Training on all feature (Full)

To quantify the performance of each configuration, we use the standard metrics precision, recall, and F1-measure. In our context, *precision* for a particular class is given by the number of tasks that were correctly assigned to this class divided by the total number of tasks that were assigned to this class. *Recall* is given by the number of tasks that were correctly assigned to this class divided by the total number of tasks belong to that class. The *F1-measure* is the harmonic mean of the two. Note that precision, recall, and F1-measure are computed for each class individually. To also provide aggregate results, we conduct micro averaging. That is, we use the number of tasks belonging to a particular class to weight the respective precision and recall values. A macro perspective (i.e. applying no weights) would provide a distorted picture because the three classes vary in size.

4.2 Results

The results of the 10-fold cross validation are presented in Table 2. Besides precision, recall, F1-measure for each class and configuration, it also shows the number of correctly and incorrectly classified instances.

Table 2. Results from 10-fold cross validation

		A	A+O	A+O+RT	Full
	Correct	320	320	340	342
	Incorrect	104	104	84	82
Manual	Precision	0.75	0.75	0.80	0.81
	Recall	0.83	0.83	0.90	0.90
	F1-Measure	0.79	0.79	0.85	0.85
User	Precision	0.75	0.75	0.80	0.80
	Recall	0.67	0.67	0.70	0.70
	F1-Measure	0.71	0.71	0.75	0.75
Automated	Precision	0.82	0.82	1.00	0.92
	Recall	0.43	0.43	0.43	0.52
	F1-Measure	0.56	0.56	0.60	0.66
Total (mic.)	Precision	0.75	0.75	0.80	0.81
	Recall	0.75	0.75	0.80	0.80
	F1-Measure	0.75	0.75	0.80	0.81

In general, the results from Table 2 reveal that our approach works well. Out of the 424 task instances, our approach classified 342 correctly. This yields an overall F1-measure of 0.81. Taking a look at the contribution of the individual features shows that the action feature is of particular importance. Apparently the discriminating power of the action feature is considerable, already resulting in an overall F1-measure of 0.75. We can further see that adding the object feature has no effect at all. However, the resource type feature results in a further improvement of the overall F1-measure to 0.80. The IT domain feature has little effect, but apparently leads to the correct classification of at least two additional task instances.

Analyzing the results for individual classes in more detail shows that there are quite some differences among the classes. Most notably, the F1-measure for the *automated* class (0.66) is much lower than the F1-measure of the *manual* (0.85) and the *user class* (0.75). This is, however, not particularly surprising when taking the class sizes into account. The automated class only contains 21 instances, which clearly makes it a minority class. It is worth noting that the rather low F1-measure mainly results from a low recall (0.52). The precision reveals that automated task are correctly classified in 92% of the cases.

To further illustrate the results, Fig. 3 shows the ROC curves and the corresponding AUC (area under the curve) values for the total configuration.[2] ROC curves are graphical representations of the proportion of true positives versus the proportion of false positives and often used to illustrate the capabilities of a binary classifier. The AUC value represents the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative one. The AUC value varies between 0 and 1. An uninformative classifier yields an AUC value of 0.5, a perfect classifier respectively yields an AUC value of 1.0. The AUC values for our approach (ranging from 0.75 to 0.78 depending on the class) indicate that our approach represents a classifier with a good performance.

To get insights into the limits of our approach, we conducted an error analysis. More specifically, we investigated which task instances were classified incorrectly and why. In essence, we observed two main types of misclassifications: (1) misclassifications due to a deviating use of feature attributes and (2) misclassifications due to insufficient training data. The first category relates to instances that were classified erroneously because the feature attributes are typically associated with another class. As an example, consider the manual task "*attach the new sct document*". For this task, our approach misses the fact that the "*sct document*" is actually a physical document. It classifies it as a user task because the verb "*send*" is often associated with user tasks in our dataset (e.g. consider "*send e-mail*"). Another example is the user task "*check notes*", which our approach classified as a manual task. Here it did not recognize the context of an information system and bases its decision on the verb "*check*" and the object "*notes*", which are often associated with manual tasks. The second category of misclassifications relates to cases where our approach erroneously classified a task because it has not seen enough training data. For example, consider the user task "*transmit a response comment*", which our approach classified as a

[2] Note that the way Weka generates ROC curves results in only as many threshold values as there are distinct probability values assigned to the positive class. Therefore, the ROC curves from Fig. 3 are only based on three data points. This, however, does not reduce their informative value.

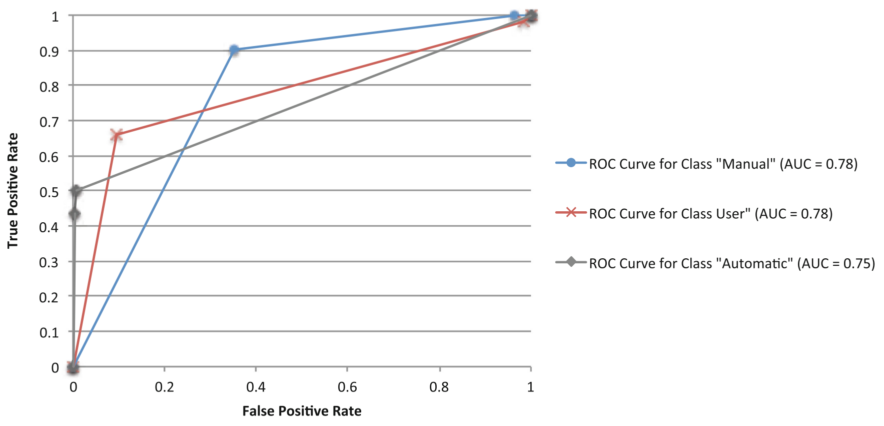


Fig. 3. ROC Curves for each class

manual task. Here the problem is that our approach has not observed a sufficient number of instances using the verb "*transmit*", which clearly relates to the use of an information system.

Despite these misclassifications, we can state that the presented approach represents a promising solution for automatically determining the degree of task automation.

5 Related Work

This paper relates to two major streams of research: (1) the application of Natural Language Process (NLP) technology in the context of business process analysis and (2) process automation.

A variety of authors have applied NLP technology in the context of business process analysis. Their works can be subdivided into techniques that analyze the natural language inside process models and techniques that analyze the natural language outside of process models, typically captured in textual process descriptions. Techniques analyzing the natural language inside process models typically focus on activity and event labels. Among others, there exist techniques for checking the correctness and consistency of process model labels [23,26,30], techniques for identifying similar parts of process models [19,25], and techniques for inferring information from process models such as service candidates [12,24]. Other approaches focus on the analysis of process-related text documents, such as approaches for the automated elicitation of process models from texts, cf. [1, 10,14] and the comparison of natural language texts to process models [2,32], and process querying based on the analysis of textual process descriptions [22].

The focus on *automation* in the context of Business Process Management is not a recent development. In particular research on workflow management and automation reaches back over 20 years [4,13,34]. Research on RPA, by contrast, is still relatively scarce. Lacity and Willcocks investigated how organizations apply RPA in practice [20,21,37]. They found that most applications of

RPA have been done for automating tasks of service business processes, such as validating the sale of insurance premiums, generating utility bills, and keeping employee records up-to date. Their study also revealed the overall potential of RPA ranging from a significant increase in turnaround times and greater workforce flexibility to cost savings of up to 30%. Other authors also studied the risks associated with BPA. For instance, Kirchmer [17] argues that RPA has the potential to make mistakes faster and with higher certainty because there is often no human check before executing an action. Davenport and Kirby also tried to answer the question of what machines are currently capable of. They argue that there are four levels of intelligence that machines can potentially master: (1) support for humans, (2) repetitive task automation, (3) context awareness and learning, and (4) self-awareness. Currently, they conclude, machines are capable of mastering level 1 and 2. Level 3 is only covered to a limited extend, level 4 not at all. They, however, stress that machines are advancing and that it is important to understand how human capabilities fit into the picture [7].

6 Conclusion

In this paper, we proposed a machine learning-based approach that automatically identifies and classifies tasks from textual process descriptions as manual, user, or automated. The goal of our technique is to reduce the effort that is required to identify suitable candidates for robotic process automation. An evaluation with 424 activities from a total of 47 textual process descriptions showed that our approach achieves an F-measure of 0.81 and, therefore, produces satisfactory results.

Despite these positive results, it is important to consider our results in the light of some limitations. First, it should be noted that the dataset we used in this paper is not representative. Textual process descriptions in practice may deviate from the ones in our dataset in different ways. However, we tried to maximize the external validity of our evaluation by choosing a dataset that combines different sources. What is more, our approach could be easily retrained on other datasets to further increase its performance. Second, our approach cannot guarantee that suitable automation candidates are identified. It rather gives an overview of the current degree of automation, which can then serve as input for further analysis.

In future work, we plan to improve the performance of our approach by including additional features and testing other classifiers. What is more, we intend to apply our approach in organizations in order to obtain feedback about its usefulness in practice.

References

1. Van der Aa, H., Leopold, H., Reijers, H.A.: Dealing with behavioral ambiguity in textual process descriptions. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 271–288. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_16

2. Van der Aa, H., Leopold, H., Reijers, H.A.: Comparing textual descriptions to process models: the automatic detection of inconsistencies. Inf. Syst. **64**, 447–460 (2017)
3. Van der Aa, H., Leopold, H., van de Weerd, I., Reijers, H.A.: Causes and consequences of fragmented process information: insights from a case study. In: Proceedings of the Annual Americas' Conference on Information Systems (2017)
4. van der Aalst, W.M.P., Barros, A.P., ter Hofstede, A.H.M., Kiepuszewski, B.: Advanced workflow patterns. In: Scheuermann, P., Etzion, O. (eds.) CoopIS 2000. LNCS, vol. 1901, pp. 18–29. Springer, Heidelberg (2000). https://doi.org/10.1007/10722620_2
5. Aguirre, S., Rodriguez, A.: Automation of a business process using robotic process automation (RPA): a case study. In: Figueroa-García, J.C., López-Santana, E.R., Villa-Ramírez, J.L., Ferro-Escobar, R. (eds.) WEA 2017. CCIS, vol. 742, pp. 65–71. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66963-2_7
6. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
7. Davenport, T.H., Kirby, J.: Just how smart are smart machines? MIT Sloan Manage. Rev. **57**(3), 21 (2016)
8. Domingos, P.: A few useful things to know about machine learning. Commun. ACM **55**(10), 78–87 (2012)
9. Dumas, M., Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
10. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 482–496. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_36
11. Fung, H.P.: Criteria, use cases and effects of information technology process automation (ITPA). Browser Download This Paper (2014)
12. Gacitua-Decar, V., Pahl, C.: Automatic business process pattern matching for enterprise services design. In: IEEE Congress on Services Part II, pp. 111–118 (2009)
13. Georgakopoulos, D., Hornick, M., Sheth, A.: An overview of workflow management: from process modeling to workflow automation infrastructure. Distrib. Parallel Databases **3**(2), 119–153 (1995)
14. Ghose, A.K., Koliadis, G., Chueng, A.: Process discovery from model and text artefacts. In: Proceedings of the IEEE Congress on Services, pp. 167–174. IEEE Computer Society (2007)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newslett. **11**(1), 10–18 (2009)
16. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026683
17. Kirchmer, M.: Robotic process automation-pragmatic solution or dangerous illusion? BTOES Insights, June 2017
18. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: 41st Meeting of the Association for Computational Linguistics, pp. 423–430 (2003)

19. Klinkmüller, C., Weber, I., Mendling, J., Leopold, H., Ludwig, A.: Increasing recall of process model matching by improved activity label matching. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 211–218. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_17
20. Lacity, M., Willcocks, L.P., Craig, A.: Robotic process automation at telefonica O2 (2015)
21. Lacity, M.C., Willcocks, L.P.: A new approach to automating services. MIT Sloan Manage. Rev. **58**(1), 41 (2016)
22. Leopold, H., van der Aa, H., Pittke, F., Raffel, M., Mendling, J., Reijers, H.A.: Searching textual and model-based process descriptions based on a unified data format. Softw. Syst. Model. 1–16 (2017)
23. Leopold, H., Eid-Sabbagh, R.H., Mendling, J., Azevedo, L.G., Baião, F.A.: Detection of naming convention violations in process models for different languages. Decis. Support Syst. **56**, 310–325 (2013)
24. Leopold, H., Mendling, J.: Automatic derivation of service candidates from business process model repositories. In: Abramowicz, W., Kriksciuniene, D., Sakalauskas, V. (eds.) BIS 2012. LNBIP, vol. 117, pp. 84–95. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30359-3_8
25. Leopold, H., Niepert, M., Weidlich, M., Mendling, J., Dijkman, R., Stuckenschmidt, H.: Probabilistic optimization of semantic process model matching. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 319–334. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32885-5_25
26. Leopold, H., Smirnov, S., Mendling, J.: On the refactoring of activity labels in business process models. Inf. Syst. **37**(5), 443–459 (2012)
27. Levin, B.: English Verb Classes and Alternations: A Preliminary Investigation. University of Chicago Press, Chicago (1993)
28. Leyh, C., Bley, K., Seek, S.: Elicitation of processes in business process management in the era of digitization – the same techniques as decades ago? In: Piazolo, F., Geist, V., Brehm, L., Schmidt, R. (eds.) ERP Future 2016. LNBIP, vol. 285, pp. 42–56. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58801-8_4
29. Miller, G., Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
30. Pittke, F., Leopold, H., Mendling, J.: Automatic detection and resolution of lexical ambiguity in process models (2015)
31. Riefer, M., Ternis, S.F., Thaler, T.: Mining process models from natural language text: A state-of-the-art analysis. In: Multikonferenz Wirtschaftsinformatik (MKWI-16). Universität Illmenau , Illmenau, Germany, 9–11 March 2016
32. Sànchez-Ferreres, J., Carmona, J., Padró, L.: Aligning textual and graphical descriptions of processes through ILP techniques. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 413–427. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_26
33. Schuler, K.K.: Verbnet: a broad-coverage, comprehensive verb lexicon. Ph.D. thesis, Philadelphia, PA, USA (2005)
34. Stohr, E.A., Zhao, J.L.: Workflow automation: overview and research issues. Inf. Syst. Front. **3**(3), 281–296 (2001)
35. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. **2**, 45–66 (2001)
36. Walenz, B., Didion, J.: JWNL: Java wordnet library (2011)
37. Willcocks, L., Lacity, M.C.: Service Automation: Robots and the Future of Work. Steve Brookes Publishing, Warwickshire (2016)

Toward an Automated Labeling of Event Log Attributes

Amine Abbad Andaloussi(✉), Andrea Burattin(✉), and Barbara Weber(✉)

DTU Compute, Software Engineering, Technical University of Denmark, 2800 Kongens Lyngby, Denmark
{amab,andbur,bweb}@dtu.dk

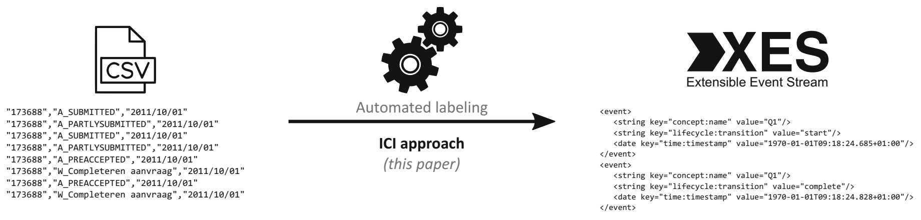
Abstract. Process mining aims at exploring the data produced by executable business processes to mine the underlying control-flow and data-flow. Most of the process mining algorithms assume the existence of an event log with a certain maturity level. Unfortunately, the logs provided by process unaware information systems often do not comply with the required maturity level, since they lack the notion of process instance, also referred in process mining as "case id". Without a proper identification of the case id attribute in log files, the outcome of process mining algorithms is unpredictable. This paper proposes a new approach that aims to overcome this challenge by automatically inferring the case id attribute from log files. The approach has been implemented as a ProM plugin and evaluated with several real-world event logs. The results demonstrate a high accuracy in inferring the case id attribute.

1 Introduction

The event logs produced by information systems provide insights into the executed process instances and allow to perceive the individual behaviour of each of them. In process mining, the control-flow is extracted from the recorded behavior which represents the order in which events were executed, and the data-flow is extracted from the correlation among events' attributes. However, to explore process mining capabilities to mine the individual behaviour of each process instance from an event log, it is necessary to distinguish and isolate each recorded process instances. This requirement gets more complicated in case of concurrent execution of process instances which is one of the fundamental principles of designing modern BPM systems [12]. Under this circumstance, the correlation among events becomes uncertain, as two successive events in the log may belong to different process instances. As solution, a case identifier (*case id*) should be attributed to each single process execution. A case identifier is assigned to events with the same attribute value for all events belonging to the same process instance.

The availability of the case id attribute in an event log depends on its level of maturity. The process mining manifesto [10] introduced a maturity ranking of event logs depending on the level of logging information they provide.

https://doi.org/10.1007/978-3-319-91704-7_6



```
"173688","A_SUBMITTED","2011/10/01"
"173688","A_PARTLYSUBMITTED","2011/10/01"
"173688","A_SUBMITTED","2011/10/01"
"173688","A_PARTLYSUBMITTED","2011/10/01"
"173688","A_PREACCEPTED","2011/10/01"
"173688","W_Completeren aanvraag","2011/10/01"
"173688","A_PREACCEPTED","2011/10/01"
"173688","W_Completeren aanvraag","2011/10/01"
```

```
<event>
   <string key="concept:name" value="Q1"/>
   <string key="lifecycle:transition" value="start"/>
   <date key="time:timestamp" value="1970-01-01T09:18:24.685+01:00"/>
</event>
<event>
   <string key="concept:name" value="Q1"/>
   <string key="lifecycle:transition" value="complete"/>
   <date key="time:timestamp" value="1970-01-01T09:18:24.828+01:00"/>
</event>
```

Fig. 1. Context of the approach presented in this paper

Among the maturity criteria used for the ranking is the notion of case id that should be explicitly mentioned in the log. Burattin in [4] characterized companies according to the process awareness of their working methodology and to the process awareness of the software system they use. In this context, several process aware companies use process unaware systems where the notion of case id is not explicit. The approach presented in this paper is beneficial for the following types of process unaware software systems: *(a)* Customer Relationship Management systems (CRM) where the process data is spread over a complex relational database; *(b)* Internet of Things (IoT) environments where sensor data is recorded on external log files; *(c)* Document Management Systems (DMS) where digital documents are stored and managed (if the company works in a process oriented manner, metadata of each documents are likely to contain information concerning the case id, such as client number or invoice number). The event logs produced by those systems have a low maturity ranking since they all lack the notion of a case id. Moreover, the log data is usually spread over dozens of attributes, making it very impractical to manually try them all before finding a correct match with the case id attribute. As the identification of the case id attribute is crucial for most process mining analyses, an approach to infer the case id from such software system logs is valuable.

The context of the approach is depicted in Fig. 1. The purpose is to automate the labeling in the process of converting log files from CSV format to XES[1] format. This paper addresses the challenge of inferring the case id as an initial stride toward a generic approach allowing to infer all the other relevant event log attributes (i.e., activity name, resource). By exploring the control-flow discovery quality dimensions a new approach to infer the case ids from event logs (Infer Case Id, abbreviated as ICI) is introduced and evaluated using both synthetic and real-world event logs. The rest of the paper is structured as follows: Sect. 2 discusses the related work. Section 3 presents the event log labeling approach along with the concepts used to explain it. Section 4 describes the implementation. Section 5 evaluates the proposed approach, and Sect. 6 discusses the obtained results. Finally, Sect. 7 concludes the paper.

[1] See https://standards.ieee.org/findstds/standard/1849-2016.html.

2 Related Work

The challenge of inferring case ids from event logs has obtained little attention from the BPM community. The reason is that most of the literature introducing new process mining techniques assume the existence of labelled log files where the case id is known beforehand. This paper drops such assumption, and goes on a quest for automatically inferring case ids. By looking at the existing related work, it is notable that few publications [2,5,6,14] have already raised this challenge and proposed different approaches from different perspectives to solve the problem.

Ferreira and Gillblad in [6], proposed an approach to transform an unlabeled log into a labelled one using the Expectation-Maximization technique. This aims at finding a solution that converges to a local maximum of a likelihood function. This approach is considered by the authors as generic and executable in different environments. However, inferring case ids using this technique might be subject to uncertainty because the first-order Markovian model used is unable to represent some work-flow patterns such as loops and parallelism. An enhanced approach, suggested by Walicki and Ferreira [14], suggests a sequence partitioning technique. However, the proposed technique shares the same limitations as the previous one [6] since it is limited to only simple work-flow patterns, thus it does not support loops and parallelism.

Bayomie et al. in [2] proposed an approach to infer the case identifiers from unlabeled event logs. The approach requires the reference process model used to document the business process, which is often part of the documentation package delivered at design time. The reference process model is used to generate a causal behavioural profile [15]. The latter is used together with time heuristics inferred from the event log to build a decision tree where each node represents an event from the event log and carries its conditional probability of belonging to the same case as its parent node. Bayomie's approach aims at generating a set of labelled event logs listed according to a ranking score used to indicate their degree of trust. The approach explores the data-flow correlations (time heuristics) and control-flow correlations (causal behavioural profile) between events to group them by process instance.

Burattin and Vigo in [5] share the same assumption as the ICI approach, by considering the case id as a hidden attribute inside the log. The authors justify their assumption by the fact that it is general enough for a broad range of real-world event logs. Their proposed approach consists of filtering a set of event attributes considered as candidates for representing the case id in an event log. The filtering is done to reduce the search space by applying some selection heuristics such as selecting only the attributes with specific data types (i.e., ignoring timestamps), and using regular expression constraints as a selection criteria. Afterwards, the approach exploits the amount of data shared between attributes to construct chains, such that each chain links all similar attributes' values across the log. The case ids are then, inferred by choosing the chain with maximal length and minimal number of crossed attributes. By reducing the search space, the approach aims at finding a set of possible combinations of attribute that might represent the case id. However, it relies entirely on the similarity of attributes; thus, its accuracy is limited to a specific range of logs.

The ICI approach presented in this paper overcomes all the challenges of the previously cited approaches. Indeed, it does not require any reference process model nor similar heuristics to infer the case ids. The aim is to introduce a generic approach to infer the case id – but other attributes as well – from event logs using the control-flow discovery quality dimensions.

3 Method

The ICI approach automates the event log labeling process. Section 3.1 provides a formal definition of the notations used to describe the approach, Sect. 3.2 presents the control-flow quality dimensions, Sect. 3.3 highlights the underlying assumptions, Sect. 3.4 presents the approach, and Sect. 3.5 illustrates the ICI approach by providing a running example.

3.1 Preliminaries

In this section, formal definitions for *sequence*, *event*, *raw event*, *trace, case*, and *event log* are provided. These definitions are combined from existing work available in the literature [8].

Definition 1 (Sequence). *Given a set $\mathcal{A}$, a* finite sequence *over $\mathcal{A}$ of length n is a mapping $s \in ([1, n] \subset \mathbb{N}) \rightarrow \mathcal{A}$ and can be represented as a string, i.e., $s = \langle s_1, s_2, \ldots, s_n \rangle$, where $s_i \in \mathcal{A}$. We write $s \in \mathcal{A}^*$ to indicate that the sequence s contains elements from $\mathcal{A}$. Over a sequence s the following functions are defined:*

- *Selection operator: $s(i) = s_i$, $\forall\, 1 \leq i \leq n$;*
- *Size operator: $|s| = n$ (i.e., the length of the sequence).*

Definition 2 (Event). *Given any notion of process, let $\mathcal{A}$ be the set of all possible activities contained in the process, let $\mathcal{C}$ be the set of all possible case ids (i.e., the set of all possible identifiers of process instances), and let $\mathcal{D}_1, \ldots, \mathcal{D}_m$ be the set of additional data attributes characterizing each event executed. An* event *is a tuple $e = (a, c, t, d_1, \ldots, d_m)$, where:*

- *$a \in \mathcal{A}$ represents the activity associated to the event;*
- *$c \in \mathcal{C}$ represents the case id;*
- *$t \in \mathbb{N}$ represents the timestamp;*
- *$d_1, \ldots, d_m$ is a list of additional (and optional) event attributes, where $\forall\, 1 \leq i \leq m, d_i \in \mathcal{D}_i \cup \{\perp\}$.*

$\mathcal{E} = \mathcal{A} \times \mathcal{C} \times \mathbb{N} \times \mathcal{D}^{\perp}{}_1 \times \ldots \times \mathcal{D}^{\perp}{}_m$ is called the event universe. In an event e, the following projection functions are defined: $\pi_a(e) = a$, $\pi_c(e) = c$, $\pi_t(e) = t$, and $\pi_{d_i}(e) = d_i, \forall\, 1 \leq i \leq m$. If e does not contain the attribute value d_i for some $i \in [1, m] \subset \mathbb{N}$, $\pi_{d_i}(e) = \perp$.

The event definition (cf. Definition 2) assumes the existence of a case id $c \in \mathcal{C}$ in order to have the tuple $e = (a, c, t, d_1, \ldots, d_m)$. Since the ICI approach assumes that the case id is unknown a-priori, it is necessary to define a *Raw Event* as an event with unknown case id.

Definition 3 (Raw Event). *A* Raw Event *is a tuple* $\hat{e} = (a, t, d_1, \ldots, d_k)$*, where:*

- $a \in \mathcal{A}$ *represents the activity associated to the event;*
- $t \in \mathbb{N}$ *represents the timestamp;*
- $d_1, \ldots, d_k$ *is a list of raw event attributes.*

$\hat{\mathcal{E}} = \mathcal{A} \times \mathbb{N} \times \mathcal{D}_1 \times \ldots \times \mathcal{D}_k$ *is called the raw event universe. In a raw event* $\hat{e}$*, the following projection functions are defined:* $\pi_a(\hat{e}) = a$*,* $\pi_t(\hat{e}) = t$*, and* $\pi_{d_i}(\hat{e}) = d_i, \forall\, 1 \leq i \leq k$*. If* $\hat{e}$ *does not contain the attribute value* d_i *for some* $i \in [1, k] \subset \mathbb{N}$*,* $\pi_{d_i}(\hat{e}) = \perp$*.*

Definition 4 (Trace, Case). *A* trace *is defined as a finite sequence of events* $\sigma_c = \langle e_1, e_2, \ldots, e_n \rangle \in \mathcal{E}^*$ *such that* $\exists c \in \mathcal{C}\ \forall\, 1 \leq i \leq |\sigma|, \pi_c(e_i) = c \wedge \forall\, 1 \leq j < |\sigma_c|$*,* $\pi_t(\sigma_c(e_j)) \leq \pi_t(\sigma_c(e_{j+1}))$*. In the context of this paper, each* case *is a grouping of events belonging to the same process execution and having same case id. Thus, each* case *is a distinct* trace*. Additionally, the sequence of events in a trace is ordered according to their timestamp.*

Definition 5 (Event Log). *An* Event Log L *is defined as a set of events such that* $L \subseteq \mathcal{E}$*. Please note that it is possible to group events based on their case id in order to identify traces.*

3.2 Control-Flow Quality Dimensions

This section describes the control-flow quality dimensions considered by the ICI approach. The availability of an event log allows generating different process models depending on the discovery algorithm used. The generated models can be evaluated based on the following four quality dimensions: *Fitness*, *Precision*, *Generalization*, and *Simplicity* [12].

The Fitness dimension represents the ability of a process model to reproduce the control-flow of the traces recorded in the event log [11]. Measuring Fitness can be performed using several approaches such as the "Alignment-based Replay Fitness" [13] and the Petri-net replay technique that allows to detect possible mismatches [9]. The Precision dimension is used to quantify the extra behaviour allowed by a process model that is not recorded in the log [11]. In case the process model contains loops, this will generate infinitely many behaviours. Therefore, counting the number of all possible traces is impossible. There exist several approach to quantify precision such as the *Escaping edges* technique [3], and alignment based technique proposed in [1]. The *generalization* dimension is used to quantify the extent to which the process model can replay log traces that are not yet recorded in the event log [11]. There exist several approaches to estimate generalization such as *the frequency of use* approach, which is based on the assumption that a generic model is a model whose parts are all used with the same frequency [3]. The Simplicity dimension is used to quantify the extent to which a process model is simple. It is defined according to two main principles: (a) The Occam's Razor principle which states "One should not increase, beyond what is necessary, the number of entities required to explain anything." (b) The understandability of the process model by the user [3]. The literature presents several heuristics that could be used to estimate simplicity such as the *Simplicity by activity occurrence* [3]. This heuristic assumes that the less duplicate activities there are in a process model, the more simple it is.

3.3 Assumptions

The ICI approach is built upon few assumptions. Specifically, the case id is assumed to be explicitly mentioned in the event log. In other words, given a raw event $\hat{e} = (a, t, d_1, \ldots, d_k)$ (Definition 3), a case id c is one of the raw event attributes $d_1, \ldots, d_k$. Additionally, the case id is given by the same attribute of all raw events. In other words, if d_i is the case id attribute of raw event $\hat{e} \in \hat{\mathcal{E}}$, then the case id attribute of all other events in $\hat{\mathcal{E}}$ is d_i. Finally, the event name attribute and the timestamp attribute of the event log are know and each raw event set refers to executions of the same process (with several instances).

Please note that all these assumptions are typically acceptable in many real process mining projects.

3.4 Approach

The preliminaries presented in Sect. 3.1, the four quality dimensions described in Sect. 3.2, and the assumptions presented in Sect. 3.3 provide a good starting point to describe the ICI approach. The control-flow discovery allows discovering process models reflecting the behaviours seen in an event log [11, p. 125]. To ensure the consistency of the discovered model, the *case id* should be correctly identified in the log. In case it is wrongly identified, the obtained model would be inconsistent. For instance, by selecting a random attribute as a case id it is most likely that the discovered control-flow would not represent the original process model. Consequently, the discovery algorithm used will produce some strange behaviours resulting in an inconsistent model. Luckily, the control-flow four quality dimensions allow quantifying those behaviours.

In principle, the control-flow discovery quality dimensions are meant to evaluate and compare the quality of different discovery algorithms [3]. This paper goes beyond the classical use of the control-flow discovery quality dimensions by exploiting their ability to evaluate and compare different process models obtained using the *same* process discovery algorithm but considering different event attributes as case id.

To Infer the case id attribute from a log file, the ICI approach relies on two important steps. First it uses a heuristic to quantify the number of distinct values of each attribute across the log. This step is called *Compute grouping ratio for* *each attribute*. Its purpose is to identify the attributes that are most likely to represent the case id. In the second step, each log attribute is assumed to be the case id attribute and then evaluated using the control-flow discovery quality dimensions. This step is called *Compute the quality score for each attribute*. Using the heuristic from the first step and the evaluation metrics from the second step, the ICI approach allows to infer the real case id attribute. The remaining of this section (Sect. 3.4) explains these two steps.

Compute Grouping Ratio for Each Attribute: To get close insight into which event attribute is more likely to represent the case id, the ICI approach computes the *grouping ratio* for each attribute. This is used to quantify the extent to which an attribute can be used to split events into groups, such that each group can be identified by a unique value of the attribute. For instance, a case id attribute is used to group events belonging to the same process execution by assigning them the same value, thus by grouping events by case id, the obtained number of groups will correspond to the number of process executions (cases). However, by choosing a different attribute to represent the case id (i.e., timestamp, event id, resource), the obtained event groups might have smaller or larger size. Section 3.5 provides an example showing the intuition behind the use of the grouping ratio to measure the likelihood that an attribute is the case id.

Let $\hat{L}$ be a set of raw events such that $\hat{L} \subseteq \hat{\mathcal{E}}$. According to Definition 3, a raw event $\hat{e} \in \hat{L}$ is a tuple $\hat{e} = (a, t, d_1, \ldots, d_k)$, with $d_1, \ldots, d_k$ being the list of raw event attributes, where $\forall\, 1 \leq i \leq k, d_i \in \mathcal{D}_i$. $\mathcal{N}_i$ is defined as the set of unique values for $\mathcal{D}_i$ such that $\mathcal{N}_i(\hat{L}) = \{\pi_{d_i}(\hat{e}) \mid \hat{e} \in \hat{L}\}$. Then, the *Grouping Ratio* for $\mathcal{D}_i$ on the raw events set $\hat{L}$ is defined as:

$$g_i(\hat{L}) = 1 - \frac{|\mathcal{N}_i(\hat{L})|}{|\hat{L}|}. \tag{1}$$

Where $|\mathcal{N}_i(\hat{L})|$ is the size of set $\mathcal{N}_i(\hat{L})$, and $|\hat{L}|$ is the size of the set $\hat{L}$ that is the total number of raw events it contains.

Compute Quality Score for Each Attribute: In this step, *raw events* are transformed into *events* with known case id that is one of the event attributes, and then an event log L is generated. Afterwards, a process discovery algorithm is applied to the event log L to discover the corresponding process model. Once the model is obtained, the quality dimension metrics are used to measure fitness, precision, generalization, and simplicity. The measurements are summed up together with the *distance to the average grouping ratio* then averaged to get a *quality score*, which is used to rank the process model corresponding to each attribute. Finally, the attribute with the highest rank is selected to be the real case id attribute.

Let $\hat{\mathcal{L}}$ be the set of raw events (cf. Definition 3) and let L be an event log (cf. Definition 5). $\hat{\mathcal{L}}$ can be transformed to L as follows: Let k be the number of the of additional data attributes $\mathcal{D}_1, \ldots, \mathcal{D}_k$ and let $j \in [1, k]$ be the index of the attribute considered as case id. Then, for each $\hat{e} \in \hat{\mathcal{L}}$, a new event e is created such that $\pi_a(e) = \pi_a(\hat{e})$, $\pi_t(e) = \pi_t(\hat{e})$, $\pi_{d_i}(e) = \pi_{d_i}(\hat{e}), \forall\, 1 \leq i \leq k$, and $\pi_c(e) = \pi_{d_j}(\hat{e})$. Finally e is inserted into the event log L. Also, Let $g(\hat{\mathcal{L}})$ be the average grouping ratio of the additional data attributes $\mathcal{D}_1, \ldots \mathcal{D}_k$ such that $g(\hat{\mathcal{L}}) = \frac{\sum_{i \in [1,k]} g_i(\hat{\mathcal{L}})}{k}$. Then the *distance to the average grouping ratio* for an additional data attribute $\mathcal{D}_i$ is defined as $gr(\hat{\mathcal{L}}, i) = 1 - |g_i(\hat{\mathcal{L}}) - g(\hat{\mathcal{L}})|$, where $|.|$ is an absolute value function.

Algorithm 1 describes the function used to infer the case id. The function takes as input $\hat{\mathcal{L}}$ the set of raw events, and returns c the index of the case id in the list of raw event attributes. The algorithm iterates over all the indexes of the additional attributes (lines 3–16). For each index $i \in [1, k]$, it computes the distance to the average grouping ratio gr using the function $\mathsf{gr}(\hat{\mathcal{L}}, i)$ (line 4), then generates an event log file L using the function $\mathsf{generateLog}(\hat{\mathcal{L}}, i)$ where D_i is assumed to be the case id attribute (line 5), and applies a process discovery algorithm (i.e., Inductive Miner) to generate the corresponding process model M using the function $\mathsf{mine}(L)$ (line 6). Afterwards, it computes the fitness, precision, generalization, and simplicity using the functions $\mathsf{fitness}(M, L)$, $\mathsf{precision}(M, L)$, $\mathsf{generalization}(M, L)$, and $\mathsf{simplicity}(M, L)$ respectively (lines 7–10). Finally it computes the quality score *qual* from the previous quality dimensions and the distance to the average grouping ratio (line 11). Moreover, the algorithm keeps track of the attribute index with highest quality score to return it by the end of all the iterations (lines 12–15).

3.5 Running Example

To illustrate the ICI approach a synthetic log file entitled *Robot Process*[2] recording the workflow of a robot process in a smart factory is used. The important attributes in this log file are the *Case Id*, the *Start Timestamp*, the *Event Name*, and the *Subject Id* that refers to the resource attribute. To demonstrate the ICI approach, the *Case Id* which is the ground truth in this example is assumed to be unknown, and the aim is to infer it as explained in Sect. 3.4. The first step is to use Eq. 1 to calculate the grouping ratio of each event attribute (log column) in the log file. The results are shown in Table 1.

The grouping ratios presented in Table 1 provide a brief insight into which event attributes might represent the case id. By analyzing the obtained grouping ratios, one can notice the following: *Event Id* attribute has score 0, which is trivial since the event Id is a unique identifier for each event. *Start Timestamp* and *End Timestamp* have very low grouping ratios because some events happened concurrently. However, *Event Name*, *Event type*, *Subject Group*, and *Object Group* have very high grouping ratios. Assuming that a case id should not group too many nor too few events one would expect that only *Case Id*, *Subject Id*, and *Object Id* might represent the real case id column.

The second step is to compute the quality score for the attributes in the set of raw events as shown in Table 2. In this step, each attribute is considered as

[2] See https://doi.org/10.5281/zenodo.1186684.

Algorithm 1. Infer case id

```
Input  : $\hat{\mathcal{L}}$ the set of raw events, where $\mathcal{D}_1, \ldots, \mathcal{D}_k$ are all additional data
         attributes available
Output: $c$ the index of the case id in the list of raw event attributes
1  $best \leftarrow 0$                                  // Initialize highest score
2  $c \leftarrow \perp$
   // Iterate over all the indexes of the additional attributes
      $\mathcal{D}_1, \ldots, \mathcal{D}_k$
3  foreach $i \in [1, k]$ do
4      $gr \leftarrow \mathsf{gr}(\hat{\mathcal{L}}, i)$ // Compute the distance to the average grouping ratio
5      $L \leftarrow \mathsf{generateLog}(\hat{\mathcal{L}}, i)$        // Generate log file using $D_i$ as case id
6      $M \leftarrow \mathsf{mine}(L)$                   // Apply process discovery algorithm
       // Compute all quality dimensions
7      $f \leftarrow \mathsf{fitness}(M, L)$
8      $p \leftarrow \mathsf{precision}(M, L)$
9      $g \leftarrow \mathsf{generalization}(M, L)$
10     $s \leftarrow \mathsf{simplicity}(M, L)$
11     $qual \leftarrow (gr + f + p + g + s)/5$               // Compute quality score
12     if $qual > best$ then
13         $c \leftarrow j$   // Consider $j$ as the index of the new candidate case id
14         $best \leftarrow qual$
15     end
16 end
17 return $c$
```

Table 1. Grouping ratios for *Robot Process* log attributes

CaseId	EventId	StartT	EndT	E.Name	E.Type	S.Group	S.Id	O.Group	O.Id
0.7693	0.0000	0.0079	0.1696	0.9992	0.9997	0.9997	**0.9861**	0.9997	**0.9333**

candidate case id. For the sake of simplicity it is assumed that the event name attribute and the start timestamp attribute are known; thus they are excluded from the set of possible attributes. For each candidate case id attribute, the corresponding process model is generated using a process discovery algorithm. In this paper, the "Inductive Miner with Infrequent and all operators (IMfa)" [7] is used. IMfa is considered as one of the least biased discovery algorithms toward the four quality dimensions. Then, the distance to the average grouping ratio *(Gr)* is calculated from the grouping ratios *(G)*. Afterwards, the control-flow discovery quality dimension metrics *(Fr, Pi, Sm, and Gv)* are computed for each attribute and the quality scores *(Quality)* are derived. In this paper, the control-flow discovery quality dimensions are calculated using the metrics presented in [3]: the fitness, the precision, the generalization and the simplicity are calculated using *Alignment-based Replay Fitness*, *Escaping edges*, *Frequency of use*, and *Simplicity by activity occurrence* metrics respectively.

Table 2. Quality scores for each candidate case id

Attribute	G	Gr	Fr	Pi	Sm	Gv	Quality	Rank
Case Id	0.7693	0.9172	0.9998	1.0000	1.0000	0.9726	**0.9779**	**1**
Event Id	0.0000	0.3136	1.0000	1.0000	1.0000	0.9747	0.8577	5
End Time stamp	0.1696	0.4832	0.9543	0.9171	1.0000	0.9767	0.8663	4
Event Type	0.9997	0.6867	0.7975	0.4927	1.0000	0.8450	0.7644	6
Subject Group	0.9997	0.6867	0.7975	0.4927	1.0000	0.8450	0.7644	6
Subject Id	0.9861	0.7004	0.9997	0.8077	1.0000	0.9596	0.8935	2
Object Group	0.9997	0.6867	0.7975	0.4927	1.0000	0.8450	0.7644	6
Object Id	0.9333	0.7532	0.9991	0.7343	1.0000	0.9738	0.8921	3

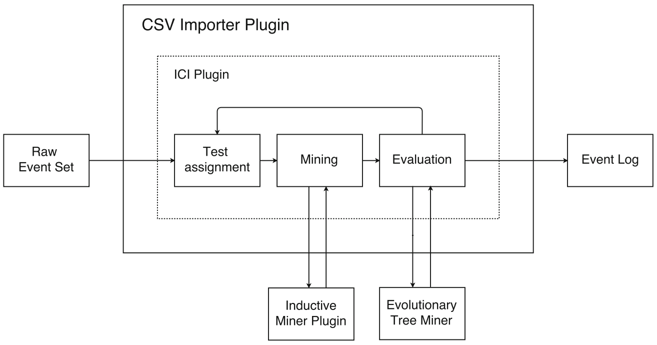


Fig. 2. ICI plugin Architecture

By ranking the obtained quality scores shown in Table 2, the *Case Id* attribute represents the best candidate as real case id because it has the highest quality score. This example illustrates the ICI approach and demonstrates its ability to provide accurate results on a synthetic log file. The next section evaluates the ICI approach on real-world event log files.

4 Implementation

An overview of the ICI plugin architecture is depicted in Fig. 2. The ICI plugin requires as input a *Raw Event Set* in the CSV format, and an initial mapping with the timestamp and event name attributes. The aim is to infer the case id attribute among the log attributes using the approach introduced in Sect. 3. For this purpose, the followings three components have been implemented: *(a) Test assignment* which iterates over the log attributes, and constructs an event log where the case id corresponds to one of the log attributes (cf. Algorithm 1, Line 5); *(b) Mining* which provides the event log to a mining algorithm and returns the corresponding model (cf. Algorithm 1, Line 6); *(c) Evaluation* which evaluates the model using the control-flow discovery quality dimensions (cf. Algorithm 1, Lines 7–10). By end of this process, the event log with the highest quality score is chosen.

The ICI plugin integrates two main packages from the open-source process mining framework ProM[3] that are the *Inductive Miner*[4] which implements the IMfa algorithm, and the *Evolutionary Tree Miner*[5] which implements the necessary metrics used to evaluate the control-flow four quality dimensions. The ICI plugin is embedded with the CSV Importer Plugin[6].

As mentioned in Sect. 3.4 choosing the wrong attribute to represent the case id causes inconsistencies with the mining algorithm. These inconsistencies are amplified when choosing an attribute with too low or too high grouping ratio. As solution, the ICI plugin allows choosing a sample of the event log instead of using the full event log to infer the case id, which reduces the plugin execution time and avoids memory overheads due to the inconsistencies in the mining algorithm. A sample can be selected from the top n entries of the log or from a random selection of n entries in the log.

The ICI plugin (called "Infer Case ID" in ProM) is available as part of the CSV Importer package[7]. It can be installed using the ProM Package Manager. A video demonstration of the plugin is available at https://youtu.be/OKyuc3mEG1I.

5 Evaluation

To evaluate the ICI approach on a larger scale, several real-world event logs were obtained from the 4TU public database[8]. With the purpose of having a ground truth to evaluate the accuracy of ICI approach, the event logs used are all *labelled*: the real case id attribute is known. This section reports the results obtained by applying the ICI approach on several real-world log files. Section 5.1 explains the evaluation procedure and presents the used data sets, and Sect. 5.2 reports the evaluation results.

5.1 Evaluation Procedure and Data Sets

Most of the event logs obtained from the 4TU database are available in XES format, thus, the attribute labels are already known. The process mining tool Disco[9] was used to convert the event logs from XES to CSV format. The sample used in the evaluation consists of the top 10000 events ordered by timestamp (in an ascending order) for each event log. The plugin was executed with the following system configuration: 12 Gb of RAM, and 1 processor with 4 cores.

[3] See http://www.promtools.org/.
[4] See https://svn.win.tue.nl/repos/prom/Packages/InductiveMiner/.
[5] See https://svn.win.tue.nl/repos/prom/Packages/EvolutionaryTreeMiner/.
[6] See https://svn.win.tue.nl/repos/prom/Packages/CSVImporter/.
[7] Currently available in ProM Nightly Build at http://www.promtools.org/doku.php?id=nightly.
[8] See the collection of real-world event logs at 4TU Center for Research Data http://data.4tu.nl/repository/collection:event_logs_real.
[9] See https://fluxicon.com/disco/.

The data sets considered to evaluate the ICI approach are the following: BPI challenge 2012, BPI challenge 2013 incidents, BPI challenge 2014 Detail Incident Activity, BPI challenges 2017, Credit requirements, Helpdesk anonymized[10] and Receipt phase of an environmental permit application process (WABO) CoSeLoG project. These event logs are available in the Real Event Logs collection[11] of the 4TU database.

5.2 Results

The evaluation results are reported in Table 3. For each log file event attribute the following ratios are computed: grouping ratio for the full log file *(G full)*, grouping ratio of the sample used in the evaluation *(G sample)*, distance to average grouping ratio of the sample *(Gr)*, alignment-based replay fitness *(Fr)*, precision using escaping edges improved technique *(Pi)*, generalization using frequency of use technique *(Gv)*, simplicity using activity occurrence technique *(Sm)* and quality score *(Quality S.)*. The obtained quality scores are ranked to infer the log attribute with the highest rank. Note that the quality score calculation considers the grouping ratio of the sample instead of the grouping ratio of the full log. However, both grouping ratios (sample and full) are reported to emphasis on the fact that the grouping ratio of the sample used does not differ much from the grouping ratio of the original log file.

To demonstrate the accuracy of the ICI approach, the real case id attribute (considered as ground truth) should always have the highest rank among the other attributes. For sake of brevity, only the top three attributes with the highest rank are reported in Table 3. The evaluation results show that the case id attribute always has the highest quality score. The results are discussed in details in Sect. 6. The complete version of the evaluation results including the quality scores of all the attributes considered for each event log is available at https://doi.org/10.5281/zenodo.1186678.

6 Discussion

This section discusses the evaluation of the ICI approach based on the data shown in Table 3. Clearly, the ICI approach demonstrates a high accuracy in inferring the case id in all the event logs considered for the evaluation. However, it is still important to highlight few cases where the quality score of other event attributes is very close to the case id attribute score. For instance, in BPI challenge 2013, the quality scores for *Case ID* attribute and *Resource* attribute are 0.8834 and 0.8745 respectively. To explain this small difference in the quality scores, the process mining tool Disco was used. By inspecting the statistics provided by the tool for the process model where the case id corresponds the real case id attribute, the number of cases and resources are 954 and 776 respectively, which can also be noticed from the grouping ratios of *Case ID* attribute

[10] See https://data.mendeley.com/datasets/nm9xkzhpm4/1.

[11] See http://data.4tu.nl/repository/collection:event_logs_real.

Table 3. Quality scores of the three top ranked attributes of each event log. (refer to Sect. 5.1 for the full names the event logs.)

Log file	Attribute	G full	G sample	Gr	Fr	Pi	Sm	Gv	Quality S.	Rank
BPI challenge 2012	**Case ID**	0.9501	0.9210	0.9267	0.9097	0.7346	1.0000	0.9170	**0.8976**	**1**
	(case) AMOUNT_REQ	0.9976	0.9881	0.8596	0.9998	0.4074	1.0000	0.9076	0.8349	2
	concept:name	0.9999	0.9976	0.8501	0.9109	0.6615	1.0000	0.5169	0.7879	3
BPI challenge 2013	**Case ID**	0.8847	0.9045	0.9993	0.9926	0.4574	1.0000	0.9678	**0.8834**	**1**
	Resource	0.9780	0.9223	0.9815	0.9910	0.4358	1.0000	0.9640	0.8745	2
	impact	0.9999	0.9996	0.9042	0.9588	0.6001	1.0000	0.8457	0.8618	3
BPI challenge 2014	**Incident ID**	0.9471	0.9471	0.8049	0.9977	0.3541	1.0000	0.8922	**0.8098**	**1**
	IncidentActivity _Number	0.0000	0.0000	0.2480	1.0000	1.0000	1.0000	0.7271	0.7950	2
	Assignment Group	0.9886	0.9886	0.7634	0.9861	0.2568	1.0000	0.8846	0.7782	3
BPI challenges 2017	**Case ID**	0.9439	0.9084	0.9246	0.9806	0.7591	1.0000	0.9370	**0.9203**	**1**
	(case) RequestedAmount	0.9988	0.9901	0.8429	0.9858	0.5070	1.0000	0.9489	0.8569	2
	EventID	0.0000	0.0912	0.2582	1.0000	1.0000	1.0000	0.9135	0.8343	3
Credit requirements	**Case ID**	0.8750	0.8712	0.7327	0.9916	1.0000	1.0000	0.9718	**0.9392**	**1**
	Complete Timestamp	0.0127	0.0123	0.4084	0.9983	0.8900	1.0000	0.9821	0.8558	2
	Resource	0.9999	0.9992	0.6047	0.9442	0.6677	1.0000	0.4905	0.7414	3
Helpdesk anonymized	**Case ID**	0.8168	0.8180	0.8870	0.9643	0.9729	1.0000	0.9494	**0.9547**	**1**
	customer	0.9832	0.9727	0.9583	0.9302	0.5364	1.0000	0.8786	0.8607	2
	product	0.9990	0.9984	0.9326	0.9401	0.4979	1.0000	0.8761	0.8493	3
Receipt env. Permit	**Case ID**	0.8327	0.8327	0.9850	0.9843	0.7017	1.0000	0.8838	**0.9110**	**1**
	(case) responsible	0.9953	0.9953	0.8223	0.9967	0.2582	1.0000	0.8714	0.7897	2
	org:group	0.9987	0.9987	0.8190	0.9795	0.3507	1.0000	0.7992	0.7897	3

and *Resource* attribute that are 0.9045 and 0.9223 respectively. This small difference in the quality score can be explained with the fact that the sample of the event log used considers only the top 10000 events which is not enough to perceive the overall behaviour of the model. Alternatively, a large sample of 40000 events is applied to the BPI challenge 2013 event log. The corresponding quality scores are shown in Table 4.

As shown in Table 4, by increasing the event log sample size, the difference between the quality scores for *Case ID* attribute and *Resource* attribute increased significantly (0.9390 and 0.8273 respectively). In a perfect scenario, one would use the full event log to preserve its overall behaviour. However, memory overhead issues might happen especially while dealing with large event logs.

BPI challenge 2014 represents another example where the difference in quality scores between *Case ID* attribute and *IncidentActivityNumber* attribute is insignificant (0.8098, and 0.7950 respectively). However, the grouping ratio of *IncidentActivityNumber* attribute is 0; hence, the attribute contains only unique values. Consequently, the generated model would appear like a flower model

Table 4. Quality scores for BPI challenge 2013 with a sample size of 40000 events

Log file	Attribute	G sample	Gr	Fr	Pi	Sm	Gv	Quality	Rank
BPI Chal. 2013 (40000 events)	**Case ID**	0.8666	0.9624	0.9641	0.8205	1.0000	0.9480	**0.9390**	**1**
	Resource	0.9674	0.9368	0.9001	0.4056	1.0000	0.8938	0.8273	2
	product	0.9846	0.9196	0.9996	0.2387	1.0000	0.9709	0.8258	3

where all the log traces can be replayed, thus, the fitness will certainly be equal to 1. [11, p. 151]. To avoid such cases, the attributes with a grouping ratio equals to 0 could be filtered out before running the ICI plugin.

The evaluation results demonstrate that the ICI approach is accurate on several event logs. However, more event logs should be tried-out and other control-flow discovery algorithms should be used. Nevertheless, the results shown in Sect. 5.2 are promising and provides a clear insight into the aspects that should be enhanced. Mainly, the following challenges should be addressed: *(a)* Filtering out the log attributes with too low or too high grouping ratios. By overcoming this challenge, the memory overhead issues will be avoided. Moreover, the attributes with Boolean values will be ignored. Such attributes always have higher grouping ratio, which impacts negatively on the distance to average grouping ratio in case several log attributes are Booleans. *(b)* Defining an optimal sample size proportional to each event log characteristics (i.e., number of resources). Indeed, The accuracy of the ICI approach depends on the sample of the log used to compute the quality score and its ability to preserve the overall behaviour, which can be quantified by the control-flow discovery quality dimensions and the grouping ratio. Please note that the sample size used in this evaluation has been chosen to preserve the overall behaviour of the log. However, it cannot guarantee that the same sample size is valid for all other log files.

7 Conclusion and Future Work

To sum up, the ICI approach proposes a new technique to automatically infer the case id from an event log by exploring the control-flow discovery quality dimensions capabilities. Unlike the existing techniques mentioned in Sect. 2, the ICI approach does not require any domain-specific heuristics. Under the assumption that a case id is explicitly mentioned in the event log, the ICI approach allowed to correctly identify the case id in a synthetic log. The approach was evaluated using several real-world event logs obtained from a public database to demonstrate its accuracy on a large scale. The results show a high potential for inferring the case id despite the challenges discussed in Sect. 6.

As future work, the challenges related to memory overhead and optimal sample size have the highest priority. Moreover. several heuristics could be applied to filter out the candidate event attributes based on their data types (i.e., ignoring timestamp attributes and Boolean attributes) and based on their grouping ratios. In addition, the availability of domain knowledge will help to reduce the search space and enhance the performance of the ICI approach by enabling a pre-selection of the event attributes that are most likely to contain the case id. Furthermore, the ICI approach could be easily generalized to infer the case id from a combination of log attributes. In term of feasibility, the approach could be illustrated in a practical use-case fitting into one of the application areas mentioned in Sect. 1. Finally, the ICI algorithm could be tried-out using other control-flow discovery algorithms and quality metrics.

References

1. Adriansyah, A., Munoz-Gama, J., Carmona, J., van Dongen, B.F., van der Aalst, W.M.P.: Measuring precision of modeled behavior. IseB **13**(1), 37–67 (2015)
2. Bayomie, D., Helal, I.M.A., Awad, A., Ezat, E., ElBastawissi, A.: Deducing case IDs for unlabeled event logs. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 242–254. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_20
3. Buijs, J.C.A.M.: Flexible evolutionary algorithms for mining structured process models (2014)
4. Burattin, A.: Process Mining Techniques in Business Environments. LNBIP, vol. 207. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17482-2
5. Burattin, A., Vigo, R.: A framework for semi-automated process instance discovery from decorative attributes. In: 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 176–183. IEEE, April 2011
6. Ferreira, D.R., Gillblad, D.: Discovering process models from unlabelled event logs. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 143–158. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03848-8_11
7. Leemans, S.J.J.: Robust process mining with guarantees. SIKS Dissertation Series No. 2017-12 (2017)
8. Polato, M., Sperduti, A., Burattin, A., et al.: Time and activity sequence prediction of business process instances. Computing, 1–27 (2018). https://doi.org/10.1007/s00607-018-0593-x
9. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008)
10. van der Aalst, W.M.P.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011, Part I. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28108-2_19
11. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-19345-3
12. van der Aalst, W.M.: Mediating between modeled and observed behavior: the quest for the 'right' process: keynote. In: Proceedings - International Conference on Research Challenges in Information Science (2013)
13. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **2**(2), 182–192 (2015)
14. Walicki, M., Ferreira, D.R.: Sequence partitioning for process mining with unlabeled event logs. Data Knowl. Eng. **70**(10), 821–841 (2011)
15. Weidlich, M., Polyvyanyy, A., Mendling, J., Weske, M.: Causal behavioural profiles - efficient computation, applications, and evaluation. Fundam. Inform. **113**(3–4), 399–435 (2011)

Specification-Driven Multi-perspective Predictive Business Process Monitoring

Ario Santoso[1,2](✉)

[1] Department of Computer Science, University of Innsbruck, Innsbruck, Austria
[2] Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy
ario.santoso@uibk.ac.at

Abstract. Predictive analysis in business process monitoring aims at forecasting the future information of a running business process. The prediction is typically made based on the model extracted from historical process execution logs (event logs). In practice, different business domains might require different kinds of predictions. Hence, it is important to have a means for properly specifying the desired prediction tasks, and a mechanism to deal with these various prediction tasks. Although there have been many studies in this area, they mostly focus on a specific prediction task. This work introduces a language for specifying the desired prediction tasks, and this language allows us to express various kinds of prediction tasks. This work also presents a mechanism for automatically creating the corresponding prediction model based on the given specification. Thus, different from previous studies, our approach enables us to deal with various kinds of prediction tasks based on the given specification. A prototype implementing our approach has been developed and experiments using a real-life event log have been conducted.

Keywords: Predictive business process monitoring
Prediction task specification · Automatic prediction model creation
Multi-perspective prediction

1 Introduction

Process mining [1] provides a collection of techniques for extracting process-related information from the logs of business process executions (event logs). One important area in this field is predictive business process monitoring, which aims at forecasting the future information of a running process based on the models extracted from event logs. Through predictive analysis, potential future problems can be detected and preventive actions can be taken in order to avoid unexpected situation (e.g., processing delay, SLA violations). Many techniques have been proposed for tackling various prediction tasks such as predicting the outcomes of a process [9,15,21,31], predicting the remaining processing time [2,23–25,30], predicting the future events [10,11,30], etc (cf. [5,8,11,17,18,22,27]).

In practice, different business areas might need different kinds of prediction tasks. For instance, an online retail company might be interested in predicting the processing time until an order can be delivered to the customer, while for an insurance company, predicting the outcomes of an insurance claim process would be interesting. On the other hand, both of them might be interested in predicting whether their processes comply with some business constraints (e.g., the processing time must be finished within a certain amount of time).

When it comes to predicting the outcomes of a process or predicting an unexpected behaviour, it is important to specify the desired outcomes or the unexpected behaviour precisely. For instance, in the area of customer problem management, to increase customer satisfaction as well as to promote efficiency, we might be interested in predicting the possibility of "*ping-pong behaviour*" among the Customer Service (CS) officers while handling the customer problems. However, the definition of a ping-pong behaviour could be varied. For instance, when a CS officer transfers a customer problem into another CS officer who belongs to the same group, it can already be considered as a ping-pong behaviour since both of them should be able to handle the same problem. Another possible definition would be when a CS officer transfers a problem into another CS officer who has the same expertise, and the problem is transfered back into the original CS officer.

To have a suitable prediction service for our domain, we need to understand and specify the desired prediction tasks properly. Thus, we need a means to express the specification. Once we have characterized the prediction objectives and are able to express them properly, we need a mechanism to create the corresponding prediction model. To automate the prediction model creation, the specification should be machine processable. As illustrated above, such specification mechanism should also allow us to specify some constraints over the data, and compare some data values at different time points. For example, to characterize the ping-pong behaviour, one possibility is to specify the behaviour as follows: "*there is an event at a certain time point in which the CS officer is different with the CS officer in the event at the next time point, but both of them belong to the same group*". Note that here we need to compare the information about the CS officer names and groups at different time points.

In this work, we tackle those problems by providing the following contributions: *(i)* We introduce a rich language for expressing the desired prediction tasks. This language allows us to specify various kinds of prediction tasks. In some sense, this language also allows us to specify how to create the desired prediction models based on the event logs. *(ii)* We devise a mechanism for building the corresponding prediction model based on the given specification. Once created, the prediction model can be used to provide predictive analysis service in business process monitoring. *(iii)* We exhibit how our approach can be used for tackling various kinds of prediction tasks (cf. Section 3.3). *(iv)* We develop a prototype that implements our approach and enables the automatic creation of prediction models based on the specified prediction objective. *(v)* To demonstrate the applicability of our approach, we carry out experiments using a real-life event log that was provided for the BPI Challenge 2013 [29].

Roughly speaking, in our approach, we specify various desired prediction tasks by specifying how we want to map each (partial) business processes execution information into the expected predicted information. Based on this specification, we automatically train either classification or regression models that will serve as the prediction models. By specifying a set of desired prediction tasks, we can obtain multi-perspective prediction services that enable us to focus on various aspects and predict various information. Our approach is independent with respect to the classification/regression model that is used. In our implementation, to get the expected quality of predictions, the users are allowed to choose the desired classification/regression model as well as the feature encoding mechanisms (to allow some sort of feature engineering). Supplementary materials containing more explanations, examples and experiments are available at [26].

2 Preliminaries

This section provides some background concepts for the rest of the paper.

Trace, Event and Event Log. We follow the usual notion of event logs as in process mining [1]. An event log captures historical information about the execution of business processes. In an event log, each execution of a process is represented as a trace. Each trace has several events, and each event in the trace captures the information about a particular event that happens during the process execution. Events are characterized by various attributes, e.g., *timestamp* (the time at which the event occurred).

Let $\mathcal{E}$ be the *event universe* (i.e., the set of all event identifiers), and $\mathcal{A}$ be the set of *attribute names*. For any event $e \in \mathcal{E}$, and attribute name $n \in \mathcal{A}$, $\#_{\text{n}}(e)$ denotes the *value of the attribute* n of e. E.g., $\#_{\text{timestamp}}(e)$ denotes the timestamp of the event e. If an event e does not have an attribute named n, then $\#_{\text{n}}(e) = \perp$ (undefined value). A *finite sequence over* $\mathcal{E}$ *of length* n is a mapping $\sigma : \{1, \ldots, n\} \rightarrow \mathcal{E}$, and such a sequence is represented as a tuple of elements of $\mathcal{E}$, i.e., $\sigma = \langle e_1, e_2, \ldots, e_n \rangle$ where $e_i = \sigma(i)$ for $i \in \{1, \ldots, n\}$. The set of all *finite sequences* over $\mathcal{E}$ is denoted by $\mathcal{E}^*$. The *length* of a sequence σ is denoted by $|\sigma|$.

A *trace* τ is a finite sequence over $\mathcal{E}$ such that each event $e \in \mathcal{E}$ occurs at most once in τ, i.e., $\tau \in \mathcal{E}^*$ and for $1 \leq i < j \leq |\tau|$, we have $\tau(i) \neq \tau(j)$, where $\tau(i)$ refers to the *event of the trace* τ *at the index* i. Let $\tau = \langle e_1, e_2, \ldots, e_n \rangle$ be a trace, $\tau^k = \langle e_1, e_2, \ldots, e_k \rangle$ denotes the *k-length prefix* of τ (for $0 < k < n$). For example, let $\{e_1, e_2, e_3, e_4, e_5, e_6, e_7\} \subset \mathcal{E}$, $\tau = \langle e_3, e_7, e_6, e_4, e_5 \rangle \in \mathcal{E}^*$ is an example of a trace, $\tau(3) = e_6$, and $\tau^2 = \langle e_3, e_7 \rangle$. Finally, an *event log* L is a set of traces such that each event occurs at most once in the entire log, i.e., for each $\tau_1, \tau_2 \in L$ such that $\tau_1 \neq \tau_2$, we have that $\tau_1 \cap \tau_2 = \emptyset$, where $\tau_1 \cap \tau_2 = \{e \in \mathcal{E} \mid \exists i, j \in \mathbb{Z}^+ \,.\, \tau_1(i) = \tau_2(j) = e\}$.

An IEEE standard for representing event logs, called XES (eXtensible Event Stream), has been introduced in [13]. The standard defines the XML format for organizing the structure of traces, events and attributes in event logs. It also introduces some extensions that define some attributes with pre-defined meaning such as: *(i)* "*concept:name*", which stores the name of event/trace;

(ii) "*org:resource*", which stores the name/identifier of the resource that triggered the event (e.g., a person name); *(iii)* "*org:group*", which stores the group name of the resource that triggered the event.

Classification and Regression. In machine learning, a classification and regression model can be seen as a function $f : \vec{X} \rightarrow Y$ that takes some *input features/variables* $\vec{x} \in \vec{X}$ and predicts the corresponding *target value/output* $y \in Y$. The key difference is that the output range of the classification task is a finite number of discrete categories (qualitative outputs) while the output range of the regression task is continous values (quantitative outputs) [12]. Both of them are supervised machine learning techniques where the models are trained with labelled data. I.e., the inputs for the training are the pairs of input variables $\vec{x}$ and target value y. This way, the models learn how to map certain inputs $\vec{x}$ into the expected target value y.

3 Approach

Our approach for obtaining a predictive process monitoring service consists of the following main steps: *(i)* specify the desired prediction tasks and *(ii)* automatically create the prediction model based on the given specification. Once created, we can use the models to predict the future information. In the following, we elaborate these steps.

3.1 Specifying the Desired Prediction Tasks

This section explains the mechanism for specifying the desired prediction task. Here we introduce a language that is able to capture the desired prediction task in terms of the specification on how to map each (partial) trace in the event log into the desired prediction results. Such specification can be used to train a classification/regression model that will be used as the prediction model.

In our approach, the specification of a particular prediction task is specified as an analytic rule, where an *analytic rule* R is an expression of the form

$$R = \langle \mathsf{Cond}_1 \Longrightarrow \mathsf{Target}_1,\ \ldots,\ \mathsf{Cond}_n \Longrightarrow \mathsf{Target}_n,\ \mathsf{DefaultTarget} \rangle.$$

Each Cond_i in R is called *condition expression*, while Target_i and $\mathsf{DefaultTarget}$ are called *target expression* (for $i \in \{1, \ldots, n\}$). We explain and formalize how to specify a condition and target expression after providing some intuitions below.

An analytic rule R will be interpreted as a function that maps (partial) traces into the values obtained from evaluating the target expressions. The mapping is based on the condition that is satisfied by the corresponding trace. Let τ be a (partial) trace, such function R can be illustrated as follows (the formal definition will be given later):

$$R(\tau) = \begin{cases} \mathit{evaluate}(\mathsf{Target}_1) & \text{if } \tau \text{ satisfies } \mathsf{Cond}_1, \\ \quad\vdots & \quad\vdots \\ \mathit{evaluate}(\mathsf{Target}_n) & \text{if } \tau \text{ satisfies } \mathsf{Cond}_n, \\ \mathit{evaluate}(\mathsf{DefaultTarget}) & \text{otherwise} \end{cases}$$

We will see that a target expression essentially specifies the desired prediction result or expresses the way how to compute the desired prediction result. Thus, an analytic rule R can also be seen as a means to map (partial) traces into the desired prediction results, or to compute the expected prediction results of (partial) traces.

To specify a condition expression in analytic rules, we introduce a language called First-Order Event Expression (FOE). Roughly speaking, an FOE formula is a First-Order Logic (FOL) formula [28] where the atoms are expressions over some event attribute values and some comparison operators (e.g., $=$, $\neq$, $>$). Moreover, the quantification in FOE is restricted to the indices of events (so as to quantify the time points). The idea of condition expressions is to capture a certain property of (partial) traces. To give some intuition, before we formally define the language, consider the ping-pong behaviour that can be specified as follows:

$$\begin{aligned}\mathsf{Cond}_{\text{pp}} = \exists i.(\ & i > \mathsf{curr} \ \wedge\ \mathsf{e}[i].\ \text{org:resource} \neq \mathsf{e}[i+1].\ \text{org:resource}\ \wedge \\ & i+1 \leq \mathsf{last}\ \wedge\ \mathsf{e}[i].\ \text{org:group} = \mathsf{e}[i+1].\ \text{org:group})\end{aligned}$$

where "$\mathsf{e}[i+1].$ org:group" is an expression for getting the "org:group" attribute value of the event at the index $i+1$. The formula $\mathsf{Cond}_{\text{pp}}$ basically says that "*there exists a time point i that is bigger than the current time point (i.e., in the future), in which the resource (the person in charge) is different with the resource at the time point i + 1 (i.e., the next time point), their groups are the same, and the next time point is still not later than the last time point*". As for the target expression, some simple examples would be some strings such as "Ping-Pong" and "Not Ping-Pong". Based on these, we can create an example of analytic rule

$$R_1 = \langle \mathsf{Cond}_{\text{pp}} \Longrightarrow \text{“Ping-Pong”},\ \text{“Not Ping-Pong”}\rangle,$$

where $\mathsf{Cond}_{\text{pp}}$ is as above. In this case, R_1 specifies a task for predicting the ping-pong behaviour. In the prediction model creation phase, we will create a classifier that classifies (partial) traces based on whether they satisfy $\mathsf{Cond}_{\text{pp}}$ or not. During the prediction phase, such classifier can be used to predict whether a given (partial) trace will lead into ping-pong behaviour or not.

The target expression can be more complex than merely a string. For instance, it can be an expression that involves arithmetic operations over numeric values such as

$$\mathsf{Target}_{\text{remainingTime}} = \mathsf{e}[\mathsf{last}].\ \text{time:timestamp} - \mathsf{e}[\mathsf{curr}].\ \text{time:timestamp},$$

which computes "*the time difference between the timestamp of the last event and the current event (i.e., remaining processing time)*". Then we can create an analytic rule

$$R_2 = \langle \mathsf{curr} < \mathsf{last} \Longrightarrow \mathsf{Target}_{\text{remainingTime}},\ 0\rangle,$$

which specifies a task for predicting the remaining time, because R_2 will map each (partial) trace into its remaining processing time. In this case, we will create a regression model for predicting the remaining processing time of a given (partial) trace. Section 3.3 provides more examples of prediction tasks specification using our language.

Formalizing the Condition and Target Expressions. As we have seen in the examples above, we need to refer to a particular index of an event within a trace. To capture this, we introduce the notion of *index expression* idx defined as follows:

$$\mathsf{idx} ::= i \mid \mathit{pint} \mid \mathsf{last} \mid \mathsf{curr} \mid \mathsf{idx}_1 + \mathsf{idx}_2 \mid \mathsf{idx}_1 - \mathsf{idx}_2$$

where *(i)* i is an *index variable. (ii)* pint is a positive integer (i.e., $\mathit{pint} \in \mathbb{Z}^+$). *(iii)* last and curr are special indices in which the former refers to the index of the last event in a trace, and the latter refers to the index of the current event (i.e., last event of the trace prefix under consideration). For instance, given a k-length prefix τ^k of the trace τ, curr is equal to k (or $|\tau^k|$), and last is equal to $|\tau|$. *(iv)* $\mathsf{idx} + \mathsf{idx}$ and $\mathsf{idx} - \mathsf{idx}$ are the usual arithmetic addition and subtraction operation over indices.

The semantics of index expression is defined over k-length trace prefixes. Since an index expression can be a variable, given a k-length trace prefix τ^k of the trace τ, we first introduce a *variable valuation* ν, i.e., a mapping from index variables into $\mathbb{Z}^+$. Then, we assign meaning to index expression by associating to τ^k and ν an *interpretation function* $(\cdot)^{\tau^k}_{\nu}$ which maps an index expression into $\mathbb{Z}^+$. Formally, $(\cdot)^{\tau^k}_{\nu}$ is inductively defined as follows:

$$\begin{array}{llll}(i)^{\tau^k}_{\nu} = \nu(i) & (\mathsf{curr})^{\tau^k}_{\nu} = k & (\mathsf{idx}_1 + \mathsf{idx}_2)^{\tau^k}_{\nu} = (\mathsf{idx}_1)^{\tau^k}_{\nu} + (\mathsf{idx}_2)^{\tau^k}_{\nu} \\ (\mathit{pint})^{\tau^k}_{\nu} = \mathit{pint} \in \mathbb{Z}^+ & (\mathsf{last})^{\tau^k}_{\nu} = |\tau| & (\mathsf{idx}_1 - \mathsf{idx}_2)^{\tau^k}_{\nu} = (\mathsf{idx}_1)^{\tau^k}_{\nu} - (\mathsf{idx}_2)^{\tau^k}_{\nu}\end{array}$$

To access the value of an event attribute, we introduce *event attribute accessor*, which is an expression of the form

$$\mathsf{e}[\mathsf{idx}].\ \mathrm{attName}$$

where *attName* is an attribute name and idx is an index expression. To define the semantics of event attribute accessor, we extend the definition of our interpretation function $(\cdot)^{\tau^k}_{\nu}$ such that it interprets an event attribute accessor expression into the attribute value of the corresponding event at the given index. Formally, $(\cdot)^{\tau^k}_{\nu}$ is defined as follows:

$$(\mathsf{e}[\mathsf{idx}].\ \mathrm{attName})^{\tau^k}_{\nu} = \begin{cases} \#_{\mathrm{attName}}(e) & \text{if } (\mathsf{idx})^{\tau^k}_{\nu} = i,\ 1 \leq i \leq |\tau|, \text{ and } \ e = \tau(i) \\ \bot & \text{otherwise} \end{cases}$$

E.g., "$\mathsf{e}[i].$ org:resource" refers to the value of the attribute "org:resource" of the event at the position i.

The value of an event attribute can be either numeric (e.g., 26, 3.86) or non-numeric (e.g., "sendOrder"), and we might want to specify properties that involve arithmetic operations over numeric values. Thus, we introduce the notion of *numeric expression* and *non-numeric expression* as expressions defined as follows:

$$\begin{array}{lll}\mathsf{nonNumExp} & ::= & \mathsf{true} \mid \mathsf{false} \mid \mathsf{String} \mid \mathsf{e}[\mathsf{idx}].\,\mathrm{NonNumericAttribute} \\ \mathsf{numExp} & ::= & \mathsf{number} \mid \mathsf{idx} \mid \mathsf{e}[\mathsf{idx}].\,\mathrm{NumericAttribute} \\ & \mid & \mathsf{numExp}_1 + \mathsf{numExp}_2 \mid \mathsf{numExp}_1 - \mathsf{numExp}_2\end{array}$$

where *(i)* true and false are the usual boolean values, *(ii)* String is the usual string, *(iii)* number is real numbers, *(iv)* $\mathsf{e}[\mathsf{idx}].\,\mathit{NonNumericAttribute}$ (resp. $\mathsf{e}[\mathsf{idx}].\,\mathit{NumericAttribute}$) is event attribute accessor for accessing an attribute with non-numeric values (resp. numeric values), *(v)* $\mathsf{numExp}_1+\mathsf{numExp}_2$ and $\mathsf{numExp}_1 - \mathsf{numExp}_2$ are the usual arithmetic operations over numeric expressions.

To give the semantics for *numeric expression* and *non-numeric expression*, we extend the definition of our interpretation function $(\cdot)_{\nu}^{\tau^k}$ by interpreting true, false, String, and number as themselves (e.g., $(3)_{\nu}^{\tau^k} = 3$, $(\text{“sendOrder”})_{\nu}^{\tau^k} =$ “sendOrder”), and by interpreting the arithmetic operations as usual, i.e., for the addition operator we have

$$(\mathsf{numExp}_1 + \mathsf{numExp}_2)_{\nu}^{\tau^k} = (\mathsf{numExp}_1)_{\nu}^{\tau^k} + (\mathsf{numExp}_2)_{\nu}^{\tau^k}$$

The definition is similar for the subtraction operator. Note that the value of an event attribute might be undefined $\perp$. In this work, we define that the arithmetic operations involving $\perp$ give $\perp$ (e.g., $26 + \perp = \perp$).

We are now ready to specify the notion of *event expression* as follows:

$$\begin{array}{lll}\mathsf{eventExp} ::= & \mathsf{numExp}_1 \ \textbf{acop}\ \mathsf{numExp}_2 \mid \mathsf{nonNumExp}_1 \ \textbf{lcop}\ \mathsf{nonNumExp}_2 \\ & \mid \mathsf{eventExp}_1 \ \textbf{lcop}\ \mathsf{eventExp}_2 \mid \mathsf{true} \mid \mathsf{false}\end{array}$$

where *(i)* **lcop** stands for a logical comparison operator ($=$ or $\neq$). *(ii)* **acop** stands for an arithmetic comparison operator ($<$, $>$, $\leq$, $\geq$, $=$ or $\neq$). We interpret each logical/arithmetic comparison operator as usual (e.g., $26 \geq 3$ is interpreted as true, “receivedOrder” = “sendOrder” is interpreted as false). It is easy to see how to extend the definition of our interpretation function $(\cdot)_{\nu}^{\tau^k}$ towards interpreting event expressions, therefore we omit the details.

Finally, we are ready to define the language for specifying condition expression, namely First-Order Event Expression (FOE). An FOE formula is a First Order Logic (FOL) formula where the atoms are event expressions and the quantification is ranging over event indices. Syntactically FOE is defined as follows:

$$\varphi ::= \mathsf{eventExp} \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \forall i.\varphi \mid \exists i.\varphi$$

Where eventExp is an event expression. The semantics of FOE constructs is based on the usual FOL semantics. Formally, given a k-length trace prefix τ^k of the trace τ, and index variables valuation ν, we extend the definition of our interpretation function $(\cdot)_{\nu}^{\tau^k}$ as follows[1]:

[1] We assume that variables are standardized apart, i.e., no two quantifiers bind the same variable (e.g., $\forall i.\exists i.(i > 3)$), and no variable occurs both free and bound (e.g., $(i > 5) \wedge \exists i.(i > 3)$). As usual in FOL, every FOE formula can be transformed into a semantically equivalent formula where the variables are standardized apart by applying some variable renaming [28].

$$\begin{array}{llll}
(\neg\varphi)_{\nu}^{\tau^k} & = \textsf{true} & \text{if} & (\varphi)_{\nu}^{\tau^k} = \textsf{false} \\
(\varphi_1 \wedge \varphi_2)_{\nu}^{\tau^k} & = \textsf{true} & \text{if} & (\varphi_1)_{\nu}^{\tau^k} = \textsf{true}, \text{ and } (\varphi_2)_{\nu}^{\tau^k} = \textsf{true} \\
(\exists i.\varphi)_{\nu}^{\tau^k} & = \textsf{true} & \text{if} & \text{for some } c \in \{1, \ldots, |\tau|\}, \text{ we have } (\varphi)_{\nu[i \mapsto c]}^{\tau^k} = \textsf{true} \\
(\forall i.\varphi)_{\nu}^{\tau^k} & = \textsf{true} & \text{if} & \text{for every } c \in \{1, \ldots, |\tau|\}, \text{ we have that } (\varphi)_{\nu[i \mapsto c]}^{\tau^k} = \textsf{true}
\end{array}$$

where $\nu[i \mapsto c]$ stands for a new index variable valuation obtained from ν as follows:

$$\nu[i \mapsto c](x) = \begin{cases} c & \text{if } x = i \\ \nu(x) & \text{if } x \neq i \end{cases}$$

Intuitively, $\nu[i \mapsto c]$ substitutes each variable i with c, while the other variables are substituted the same way as ν is defined. The semantics of $\varphi_1 \vee \varphi_2$ and $\varphi_1 \rightarrow \varphi_2$ is as usual in FOL. When φ is a closed formula, its truth value does not depend on the valuation for the index variables, and we denote the interpretation of φ simply by $(\varphi)^{\tau^k}$. We also say that τ^k *satisfies* φ, written $\tau^k \models \varphi$, if $(\varphi)^{\tau^k} = \textsf{true}$.

Finally, the condition expression in analytic rules is specified as closed FOE formulas, while the target expression is specified as either numeric expression or non-numeric expression, except that target expressions are not allowed to have index variables (Thus, they do not need variable valuation).

Essentially, FOE has the following main features: *(i)* it allows us to specify constraints over the data; *(ii)* it allows us to (universally/existentially) quantify different event time points and to compare different event attribute values at different event time points; *(iii)* it supports arithmetic expressions/operations over the data.

Checking Whether a Condition Expression is Satisfied. Given a k-length trace prefix τ^k of the trace τ, and a condition expression φ (which is expressed as an FOE formula), to explain how to check whether $\tau^k \models \varphi$, we first introduce some properties of FOE formula below. Let φ be an FOE formula, we write $\varphi[i \mapsto c]$ to denote a new formula obtained by substituting each variable i in φ by c.

Theorem 1. *Given an FOE formula $\exists i.\varphi$, and a k-length trace prefix τ^k of the trace τ,*

$$\tau^k \models \exists i.\varphi \quad \textit{iff} \quad \tau^k \models \textstyle\bigvee_{c \in \{1, \ldots |\tau|\}} \varphi[i \mapsto c]$$

Proof (sketch). By the semantics definition, τ^k satisfies $\exists i.\varphi$ iff there exists an index $c \in \{1, \ldots, |\tau|\}$, such that τ^ksatisfies the formula ψ that is obtained from φ by substituting each variable i in φ with c. Thus, it is the same as satisfying the disjunction of formulas that is obtained by considering all possible substitutions of the variable i in φ (i.e., $\bigvee_{c \in \{1, \ldots |\tau|\}} \varphi[i \mapsto c]$). This is the case because such disjunction of formulas will be satisfied by τ^k when there is a formula in the disjunction that is satisfied by τ^k. $\square$

Theorem 2. *Given an FOE formula $\forall i.\varphi$, and a k-length trace prefix τ^k of the trace τ,*

$$\tau^k \models \forall i.\varphi \quad \textit{iff} \quad \tau^k \models \textstyle\bigwedge_{c\in\{1,\ldots|\tau|\}} \varphi[i \mapsto c]$$

Proof (sketch). Similar to Theorem 1, except that we use conjunctions of formulas. □

To check whether $\tau^k \models \varphi$, we perform the following three steps: (1) Eliminate all quantifiers. This can be easily done by applying Theorems 1 and 2. As a result, each variable will be instantiated with a concrete value. (2) Evaluate each event attribute accessor expression based on the event attributes in τ. From this step, we will have a formula which is constituted by only concrete values composed by logical/arithmetic/comparison operators. (3) Last, we evaluate all logical, arithmetic and comparison operators.

Formalizing the Analytic Rule. With this machinery in hand, now we can formalize the semantics of analytic rules as introduced above. Formally, given an analytic rule

$$R = \langle \mathsf{Cond}_1 \Longrightarrow \mathsf{Target}_1,\ \ldots,\ \mathsf{Cond}_n \Longrightarrow \mathsf{Target}_n,\ \mathsf{DefaultTarget}\rangle.$$

R is interpreted as a function that maps (partial) traces into the values obtained from evaluating the target expressions defined below

$$R(\tau^k) = \begin{cases} (\mathsf{Target}_1)^{\tau^k} & \text{if } \ \tau^k \models \mathsf{Cond}_1, \\ \quad\vdots & \qquad\vdots \\ (\mathsf{Target}_n)^{\tau^k} & \text{if } \ \tau^k \models \mathsf{Cond}_n, \\ (\mathsf{DefaultTarget})^{\tau^k} & \text{otherwise} \end{cases}$$

where τ^k is k-length trace prefix of the trace τ, and recall that $(\mathsf{Target}_i)^{\tau^k}$ is the application our interpretation function $(\cdot)^{\tau^k}$ to the target expression Target_i in order to evaluate the expression and get the value. Checking whether $\tau^k \models \mathsf{Cond}_i$ can be done as explained above. We also require that an analytic rule to be *coherent*, i.e., all target expressions of an analytic rule should be either only numeric or non-numeric expressions. An analytic rule in which all of its target expressions are numeric expressions is called *numeric analytic rule*, while an analytic rule in which all of its target expressions are non-numeric expressions is called *non-numeric analytic rule.*

Given a k-length trace prefix τ^k and an analytic rule R, we say that R *is well-defined for* τ^k if R maps τ^k into exactly one target value, i.e., for every condition expressions Cond_i and Cond_j in which $\tau^k \models \mathsf{Cond}_i$ and $\tau^k \models \mathsf{Cond}_j$, we have that $(\mathsf{Target}_i)^{\tau^k} = (\mathsf{Target}_j)^{\tau^k}$. The notion of well-defined can be generalized to event logs. Given an event log L and an analytic rule R, we say that R *is well-defined for* L if for each possible k-length trace prefix τ^k of each trace τ in L, we have that R is well-defined for τ^k. This condition can be easily checked for the given event log L and an analytic rule R.

Note that our notion of well-defined is more relaxed than requiring that each condition must not be overlapped, and this gives flexibility for making a specification using our language. For instance, one can specify several characteristics of ping-pong behaviour in a more convenient way by specifying several conditional-target rules (i.e., $\mathsf{Cond}_1 \Longrightarrow$ "Ping-Pong", $\mathsf{Cond}_2 \Longrightarrow$ "Ping-Pong", ...) instead of using disjunctions of these several characteristics. From now on we only consider the analytic rules that are coherent and well-defined for the event logs under consideration.

3.2 Building the Prediction Model

Given an analytic rule R and an event log L, if R is a numeric analytic rule, we build a regression model. Otherwise, if R is a non-numeric analytic rule, we build a classification model. Note that our aim is to create a prediction function that takes (partial) traces as inputs. Thus, we train a classification/regression function in which the inputs are the features obtained from the encoding of trace prefixes in the event log L (the training data). There are several ways to encode (partial) traces into input features for training a machine learning model. For instance, [14] studies various encoding techniques such as index-based encoding, boolean encoding, etc. In [30], the authors use the so-called *one-hot encoding* of event names, and also add some time features (e.g., the time increase with respect to the previous event). In general, an encoding technique can be seen as a function enc that takes a trace τ as the input and produces a set $\{x_1, \ldots, x_m\}$ of features (i.e., $\mathsf{enc}(\tau) = \{x_1, \ldots, x_m\}$).

In our approach, users are allowed to choose the desired encoding mechanism by specifying a set Enc of preferred encoding functions (i.e., $\mathsf{Enc} = \{\mathsf{enc}_1, \ldots, \mathsf{enc}_n\}$). This allows us to do some sort of feature engineering (note that the desired feature engineering approach, that might help increasing the prediction performance, can also be added as one of these encoding functions). The set of features of a trace is then obtained by combining all features produced by applying each of the selected encoding functions into the corresponding trace. In the implementation (cf. Sect. 4), we provide some encoding functions that can be selected in order to encode a trace.

The procedure for creating the prediction model takes the following three inputs: *(i)* an analytic rule R; *(ii)* an event log L; and *(iii)* a set $\mathsf{Enc} = \{\mathsf{enc}_1, \ldots, \mathsf{enc}_n\}$ of encoding functions. The steps for creating the prediction model are as follows: (1) for each k-length trace prefix τ^k of each trace τ in the event log L (where $k \in \{2, \ldots, |\tau|\}$), we do the following three steps: *(i)* we apply each encoding function $\mathsf{enc}_i \in \mathsf{Enc}$ into τ^k, and combine all obtained features (This step gives us the encoded trace prefix τ^k_{encoded}); *(ii)* we compute the expected prediction result (target value) by applying the analytical rule R to τ^k (i.e., the target value is equal to $R(\tau^k)$); *(iii)* we add a new training instance by specifying that the prediction function $\mathcal{P}$ maps the encoded trace prefix τ^k_{encoded} into the target value computed in the previous step. (2) Finally, after processing each k-length trace prefix of each trace in the event log as in the step 1, we train the prediction function $\mathcal{P}$ based on the training instances obtained from the step 1 and get the desired prediction function. A more formal explanation of this procedure can be seen in [26].

3.3 Showcase of Our Approach: Multi-perspective Predictive Analysis Service

An analytic rule R specifies a particular prediction task of interest. To specify several desired prediction tasks, we only have to specify several analytic rules, i.e., $R_1, \ldots, R_2$. Given a set $\mathcal{R}$ of analytic rules, i.e., $\mathcal{R} = \{R_1, \ldots, R_2\}$, our approach allows us to construct a prediction model for each analytic rule $R \in \mathcal{R}$. This way, we can get a *multi-perspective prediction analysis service* provided by all of the constructed prediction models where each of them focus on a particular prediction objective.

In Sect. 3.1 we have seen some examples of prediction task specification for predicting the ping-pong behaviour and the remaining processing time. In the following, we show other examples of specifying prediction task using our language.

Predicting Unexpected Behaviour. We can specify a task for predicting unexpected behaviour by first expressing the characteristics of the unexpected behaviour. The condition expression $\mathsf{Cond}_{\mathrm{pp}}$ (in Sect. 3.1) expresses a possible characteristic of ping-pong behaviour. Another possible characterization of this behaviour is shown below:

$$\begin{array}{rrl} \mathsf{Cond}_{\mathrm{pp2}} = \exists i.(& i > \mathsf{curr} & \wedge\ \mathsf{e}[i].\ \text{org:resource} \neq \mathsf{e}[i+1].\ \text{org:resource}\ \wedge \\ & i+1 \leq \mathsf{last} & \wedge\ \mathsf{e}[i].\ \text{org:resource} = \mathsf{e}[i+2].\ \text{org:resource}\ \wedge \\ & i+2 \leq \mathsf{last} & \wedge\ \mathsf{e}[i].\ \text{org:group} = \mathsf{e}[i+1].\ \text{org:group} \\ & & \wedge\ \mathsf{e}[i].\ \text{org:group} = \mathsf{e}[i+2].\ \text{org:group}) \end{array}$$

essentially, $\mathsf{Cond}_{\mathrm{pp2}}$ characterizes the condition where "*an officer transfers a task into another officer of the same group, and then the task is transfered back into the original officer*". In the event log, this situation is captured by the changes of the org:resource value in the next event, but then it changes back into the original value in the next two events, while the values of org:group remain the same. We can then specify an analytic rule for specifying the ping-pong behaviour prediction task as follows:

$$R_3 = \langle \mathsf{Cond}_{\mathrm{pp}} \Longrightarrow \text{“Ping-Pong”},\ \mathsf{Cond}_{\mathrm{pp2}} \Longrightarrow \text{“Ping-Pong”},\ \text{“Not Ping-Pong”} \rangle.$$

During the training phase, R_3 maps each trace prefix τ^k that satisfies either $\mathsf{Cond}_{\mathrm{pp}}$ or $\mathsf{Cond}_{\mathrm{pp2}}$ into the target value "Ping-Pong", and those prefixes that neither satisfy $\mathsf{Cond}_{\mathrm{pp}}$ nor $\mathsf{Cond}_{\mathrm{pp2}}$ into "Not Ping-Pong". After the training based on this rule, we get a classifier that is trained for distinguishing between (partial) traces that will and will not lead into ping-pong behaviour. This example also exhibits the ability of our language to specify a behaviour that has multiple characteristics.

Predicting Next Event. The task for predicting the next event is specified as follows: $R_4 = \langle \mathsf{curr} + 1 \leq \mathsf{last} \Longrightarrow \mathsf{e}[\mathsf{curr} + 1].\ \text{concept:name},\ \perp \rangle$. In the training phase, R_4 maps each k-length trace prefix τ^k into its next event name, because "$\mathsf{e}[\mathsf{curr} + 1].$ concept:name" is evaluated into the name of the event at the index $\mathsf{curr} + 1$ (i.e., $|\tau^k| + 1$). If $k = |\tau|$, then R_4 maps τ^k into $\perp$ (undefined).

After the training, we get a classifier that is trained to give the next event name of the given (partial) trace.

Predicting the Next Event Timestamp. This task can be specified as follows:[2]

$$R_5 = \langle \mathsf{curr} + 1 \leq \mathsf{last} \Longrightarrow \mathsf{e}[\mathsf{curr}+1].\ \text{time:timestamp},\ \perp\rangle.$$

R_5 maps each k-length trace prefix τ^k into the next event timestamp. Hence, we train a regression model that outputs the next event timestamp of the given (partial) trace.

Predicting SLA/Business Constraints Compliance. Using FOE, we can easily specify expressive SLA conditions/business constraints, and automatically create the corresponding prediction model using our approach. E.g., we can specify a constraint:

$$\begin{aligned}&\forall\ i.(\mathsf{e}[i].\ \text{concept:name} = \text{“OrderCreated”} \ \rightarrow\ \exists\ j.(j > i\ \wedge\\ &\quad \mathsf{e}[j].\ \text{concept:name} = \text{“OrderDelivered”}\ \wedge\ \mathsf{e}[i].\ \text{orderID} = \mathsf{e}[j].\ \text{orderID}\ \wedge\\ &\qquad (\mathsf{e}[j].\ \text{time:timestamp} - \mathsf{e}[i].\ \text{time:timestamp}) < 10.800.000))\end{aligned}$$

which essentially says “*whenever there is an event where an order is created, eventually there will be an event where the order is delivered and the time difference between the two events (the processing time) is less than* 10.800.000 ms (3 h)”.

4 Implementation and Experiment

As a proof of concept, by using Java and WEKA, we have implemented a prototype[3] that is also a ProM[4] plug-in. The prototype includes a parser for our language and a program for automatically processing the specification as well as building the corresponding prediction model based on the approach explained in Sects. 3.1 and 3.2. We also provide several feature encoding functions to be selected such as one hot encoding of attributes, time since the previous event, time since midnight, attribute values encoding, etc. We can also choose the desired machine learning model to be built.

Our experiments aim at showing the applicability of our approach in automatically constructing reliable prediction models based on the given specification. The experiments were conducted using the real life event log from BPI Challenge 2013 (BPIC 13) [29]. For the experiment, we use the first 2/3 of the log for the training and the last 1/3 of the log for the testing. In BPIC 13, the ping-pong behaviour among support teams is one of the problems to be analyzed. Ideally a customer problem should be solved without involving too many

[2] Note that timestamp can be represented as milliseconds since epoch (hence, it is a number).

[3] More information about the implementation architecture, the code, the tool, and the screencast can be found at http://bit.ly/predictive-analysis.

[4] ProM is an extendable framework for process mining (http://www.promtools.org).

support teams. Here we specify a prediction task for predicting the ping-pong behaviour by first characterizing a ping-pong behaviour among support teams as follows:

$$\begin{aligned}\mathsf{Cond}_{\text{ppteam}} = \exists i.(\quad & i > \mathsf{curr} \quad \wedge\ \mathsf{e}[i].\ \text{org:group} \neq \mathsf{e}[i+1].\ \text{org:group}\ \wedge \\ & i+1 \leq \mathsf{last}\ \wedge\ \mathsf{e}[i].\ \text{concept:name} \neq \text{“Queued”})\end{aligned}$$

Roughly, $\mathsf{Cond}_{\text{ppteam}}$ says that *there is a change in the support team while the problem is not being "Queued"*. We then specify the following analytic rule:

$$R_{ex1} = \langle \mathsf{Cond}_{\text{ppteam}} \Longrightarrow \text{“Ping-Pong”},\ \text{“Not Ping-Pong”}\rangle$$

that can be fed into our tool for obtaining the prediction model. For this case, we automatically generate Decision Tree and Random Forest models from that specification. We also predict the time until the next event by specifying the following analytic rule:

$$R_{ex2} = \langle \mathsf{curr} + 1 \leq \mathsf{last} \Longrightarrow \mathsf{e}[\mathsf{curr}+1].\ \text{time:timestamp} - \mathsf{e}[\mathsf{curr}].\ \text{time:timestamp}, 0\rangle$$

For this case, we automatically generate Linear Regression and Random Forest models.

We evaluate the prediction performance of each k-length prefix τ^k of each trace τ in the testing set (for $2 \leq k < |\tau|$). We use accuracy and AUC (Area Under the ROC Curve) [12] values as the metrics to evaluate the ping-pong prediction. For the prediction of the time until the next event, we use MAE (Mean Absolute Error) [12], and RMSE (Root Mean Square Error) [12] values as the metrics, and we also provide the MAE and RMSE values for the mean-based prediction (i.e., the basic approach where the prediction is based on the mean of the target values in the training data). The results are summarized in Tables 1 and 2. We highlight the evaluation for several prediction points, namely *(i)* early prediction (at the 1/4 of the trace length), *(ii)* intermediate prediction (at the 1/2 of the trace length), and *(iii)* late prediction (at the 3/4 of the trace length). The column "All" presents the aggregate evaluation for all k-length prefix where $2 \leq k < |\tau|$.

Table 1. The evaluation of predicting ping-pong behaviour among support teams

	Accuracy				AUC value			
	Early	Mid	Late	All	Early	Mid	Late	All
Decision Tree	0.82	0.67	0.87	0.77	0.76	0.69	0.63	0.75
Random Forest	0.83	0.73	0.91	0.83	0.89	0.73	0.78	0.87

The AUC values in Table 1 show that our approach is able to automatically produce reasonable prediction models (The AUC values > 0.5). Table 2 shows that all of the automatically generated models perform better than the mean-based prediction (the baseline). The experiment also exhibits that the performance of

Table 2. The evaluation of predicting the time until the next event

	MAE (in days)				RMSE (in days)			
	Early	Mid	Late	All	Early	Mid	Late	All
Linear Regression	0.70	1.42	2.64	2.07	1.04	1.87	2.99	2.77
Random Forest	0.34	1.07	1.81	1.51	1.03	2.33	2.89	2.61
Mean-based Prediction	2.42	2.33	2.87	2.70	2.44	2.40	3.16	2.90

our approach depends on the machine learning model that is generated (e.g., in Table 1, random forest performs better than decision tree). Since our approach does not rely on a particular machine learning model, it justifies that we can simply plug in different supervised machine learning techniques in order to get different/better performance. In the future we plan to experiment with deep learning approach in order to get a better accuracy. As reported by [30], the usage of LSTM neural networks could improve the accuracy of some prediction tasks. More experiments can be seen in our supplementary materials (cf. [26]).

5 Related Work

This work is related to the area of predictive analysis in business process management. In the literature, there have been several works focusing on predicting time-related properties of running processes. For instance, the works in [2,23–25] focus on predicting the remaining processing time. The works by [18,22,27] focus on predicting delays in process execution. The authors of [30] present a deep learning approach for predicting the timestamp of the next event and use it to predict the remaining cycle time. Looking at another perspective, the works by [9,15,31] focus on predicting the outcomes of a running process. The work by [15] introduces a framework for predicting the business constraints compliance of a running process. In [15], the business constraints are formulated in propositional Linear Temporal Logic (LTL), where the atomic propositions are all possible events during the process executions. Another work on outcomes prediction is presented by [21], which proposes an approach for predicting aggregate process outcomes by also taking into account the evaluation of process risk. Related to process risks, [8] proposes an approach for risks prediction. Another stream of works tackle the problem of predicting the future events of a running process (cf. [5,10,11,24,30]).

A key difference between those works and ours is that, instead of focusing on a specific prediction task, this work enables us to specify and focus on various prediction tasks. To deal with these various desired prediction tasks, we also present a mechanism that can automatically build the corresponding prediction models based on the given specification of prediction tasks.

This work is also related to the works on devising specification language. Unlike the propositional LTL, which is the basis of Declare language [20] and typically used for specifying business constraints over sequence of events (cf.

[15]), our FOE language (which is part of our rule-based specification language) allows us not only to specify properties over sequence of events but also to specify properties over the data (attribute values) of the events. Concerning data-aware specification language, the work by [3] introduces a data-aware specification language by combining data querying mechanisms and temporal logic. Such language has been used in verification of data-aware processes systems (cf. [4,6,7]). The works by [16] enrich the Declare language with data conditions based on First-Order LTL (LTL-FO). Although those languages are data-aware, they do not support arithmetic expressions/operations over the data which is absolutely needed, e.g., for expressing the time difference between the timestamp of the first and the last event. Another interesting data-aware language is S-FEEL, which is part of the Decision Model and Notation (DMN) standard [19] by OMG. Though S-FEEL supports arithmetic expressions over the data, it does not allow us to (universally/existentially) quantify different event time points and to compare different event attribute values at different event time points, which is needed, e.g., in the ping-pong behaviour.

6 Conclusion

We have introduced a mechanism for specifying the desired prediction tasks by using a rule-based language, and for automatically creating the corresponding prediction models based on the given specification. A prototype of ProM plug-in that implements our approach has been developed and several experiments using a real life event log confirmed the applicability of our approach.

Future work includes the extension of the tool and the language. One possible extension would be to incorporate aggregate functions such as `SUM` and `CONCAT`. These functions enable us to specify more tasks such as the prediction of total cost that is based on the sum of the cost attributes in all events. The `CONCAT` function could allow us to specify the prediction of the next sequence of activities by concatenating all next activities. Experimenting with other supervised machine learning techniques would be the next step as well, e.g., using deep learning approach in order to improve accuracy.

Acknowledgement. This research has been supported by the Euregio IPN12 "*KAOS: Knowledge-Aware Operational Support*" project, which is funded by the "European Region Tyrol-South Tyrol-Trentino" (EGTC) under the first call for basic research projects. The author thanks Tri Kurniawan Wijaya for various suggestions related to this work, and Yasmin Khairina for the implementation of some prototype components.

References

1. van der Aalst, W.M.P.: Process Mining. Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. van der Aalst, W.M.P., Schonenberg, M., Song, M.: Time prediction based on process mining. Inf. Syst. **36**(2), 450–475 (2011)

3. Bagheri Hariri, B., Calvanese, D., De Giacomo, G., Deutsch, A., Montali, M.: Verification of relational data-centric dynamic systems with external services. In: the 32nd ACM SIGACT SIGMOD SIGAI Symposium on Principles of Database Systems (PODS), pp. 163–174 (2013)
4. Bagheri Hariri, B., Calvanese, D., Montali, M., Santoso, A., Solomakhin, D.: Verification of semantically-enhanced artifact systems. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 600–607. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-45005-1_51
5. Breuker, D., Matzner, M., Delfmann, P., Becker, J.: Comprehensible predictive models for business processes. MIS Q. **40**(4), 1009–1034 (2016)
6. Calvanese, D., Ceylan, İİ., Montali, M., Santoso, A.: Verification of context-sensitive knowledge and action bases. In: Fermé, E., Leite, J. (eds.) JELIA 2014. LNCS (LNAI), vol. 8761, pp. 514–528. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11558-0_36
7. Calvanese, D., Montali, M., Santoso, A.: Verification of generalized inconsistency-aware knowledge and action bases (extended version). CoRR Technical report arXiv:1504.08108, arXiv.org e-Print archive (2015). http://arxiv.org/abs/1504.08108
8. Conforti, R., de Leoni, M., La Rosa, M., van der Aalst, W.M., ter Hofstede, A.H.: A recommendation system for predicting risks across multiple business process instances. Decis. Support Syst. **69**, 1–19 (2015)
9. Di Francescomarino, C., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring. IEEE Trans. Serv. Comput. **PP**(99), 1–18 (2016)
10. Di Francescomarino, C., Ghidini, C., Maggi, F.M., Petrucci, G., Yeshchenko, A.: An eye into the future: leveraging a-priori knowledge in predictive business process monitoring. In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM 2017. LNCS, vol. 10445, pp. 252–268. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65000-5_15
11. Evermann, J., Rehse, J.R., Fettke, P.: Predicting process behaviour using deep learning. Decis. Support Syst. **100**, 129–140 (2017)
12. Friedman, J., Hastie, T., Tibshirani, R.: The Elements of Statistical Learning. Springer, New York (2001). https://doi.org/10.1007/978-0-387-21606-5
13. IEEE Comp. Intelligence Society: IEEE Standard for eXtensible Event Stream (XES) for achieving interoperability in event logs and event streams. IEEE Std 1849–2016 (2016)
14. Leontjeva, A., Conforti, R., Di Francescomarino, C., Dumas, M., Maggi, F.M.: Complex symbolic sequence encodings for predictive monitoring of business processes. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_21
15. Maggi, F.M., Di Francescomarino, C., Dumas, M., Ghidini, C.: Predictive monitoring of business processes. In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 457–472. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07881-6_31
16. Maggi, F.M., Dumas, M., García-Bañuelos, L., Montali, M.: Discovering data-aware declarative process models from event logs. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 81–96. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_8

17. Metzger, A., Leitner, P., Ivanović, D., Schmieders, E., Franklin, R., Carro, M., Dustdar, S., Pohl, K.: Comparing and combining predictive business process monitoring techniques. IEEE Trans. Syst. Man Cybern. Syst. **45**(2), 276–290 (2015)
18. Metzger, A., Franklin, R., Engel, Y.: Predictive monitoring of heterogeneous service-oriented business networks: the transport and logistics case. In: Annual SRII Global Conference (2012)
19. Object Management Group: Decision Model and Notation (DMN) 1.0 (2015). http://www.omg.org/spec/DMN/1.0/
20. Pesic, M., van der Aalst, W.M.P.: A declarative approach for flexible business processes management. In: Eder, J., Dustdar, S. (eds.) BPM 2006. LNCS, vol. 4103, pp. 169–180. Springer, Heidelberg (2006). https://doi.org/10.1007/11837862_18
21. Pika, A., van der Aalst, W., Wynn, M., Fidge, C., ter Hofstede, A.: Evaluating and predicting overall process risk using event logs. Inf. Sci. **352–353**, 98–120 (2016)
22. Pika, A., van der Aalst, W.M.P., Fidge, C.J., ter Hofstede, A.H.M., Wynn, M.T.: Predicting deadline transgressions using event logs. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 211–216. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_22
23. Polato, M., Sperduti, A., Burattin, A., de Leoni, M.: Data-aware remaining time prediction of business process instances. In: 2014 International Joint Conference on Neural Networks (IJCNN) (2014)
24. Polato, M., Sperduti, A., Burattin, A., de Leoni, M.: Time and Activity Sequence Prediction of Business Process Instances. CoRR abs/1602.07566 (2016)
25. Rogge-Solti, A., Weske, M.: Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 389–403. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-45005-1_27
26. Santoso, A.: Specification-driven multi-perspective predictive business process monitoring (extended version). CoRR Technical Report arXiv:1804.00617, arXiv.org e-Print archive (2018). https://arxiv.org/abs/1804.00617
27. Senderovich, A., Weidlich, M., Gal, A., Mandelbaum, A.: Queue mining – predicting delays in service processes. In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 42–57. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07881-6_4
28. Smullyan, R.M.: First Order Logic. Springer, Heidelberg (1968). https://doi.org/10.1007/978-3-642-86718-7
29. Steeman, W.: BPI challenge 2013 (2013). https://doi.org/10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07
30. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
31. Verenich, I., Dumas, M., La Rosa, M., Maggi, F.M., Di Francescomarino, C.: Complex symbolic sequence clustering and multiple classifiers for predictive process monitoring. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 218–229. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_18

Advanced Approaches for Business Process Modeling (BPMDS 2018)

Model Consolidation: A Process Modelling Method Combining Process Mining and Business Process Modelling

Ornela Çela([✉]), Agnès Front, and Dominique Rieu

Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
{ornela.cela, agnes.front, dominique.rieu}@univ-grenoble-alpes.fr

Abstract. Analysing the current state of a process is a crucial step when improving the process. Four elements are defined during the process analysis: the objective, the process model, the indicators and the blocking points. It is important to well-define all of them when characterizing a process state, however the process model has a more fundamental influence. This model provides a map upon which is done the analysis of the process state, prediction of its evolution and simulation of the impact of the change to be undertaken for improving the process.

In this work we are proposing a strategy for the model consolidation, that combines the process discovery and business process modeling approach. This strategy aims to merge the models derived by these existing approaches in one unique model that is complete, comprehensive, aligned to the reality and useful for in-depth analysis.

This article describes the model consolidation strategy by detailing the steps to be taken and illustrating its usage into a real-life process provided from our industrial collaborator Net Invaders [19].

Keywords: Process analysis · Business process modelling and process mining

1 Introduction

Nowadays companies are operating in a rapid changing environment that demands them to continually evolve their strategic, operational and organizational aspects. This need of continual evolution is impacting also the business processes, but their evolution is time and financially consuming and internal resistance from the actors of the process can be encountered. CEFOP method was introduced in [1, 2], to guide this continual evolution by:

1. offering a full coverage of the required functionalities for analyzing, diagnosing and continually evolving the process.
2. involving and motivating the actors of the process in participating in the process evolution, by facilitating their interaction.
3. offering the possibility to transform business and operational objectives to objectives related to the process evolving.

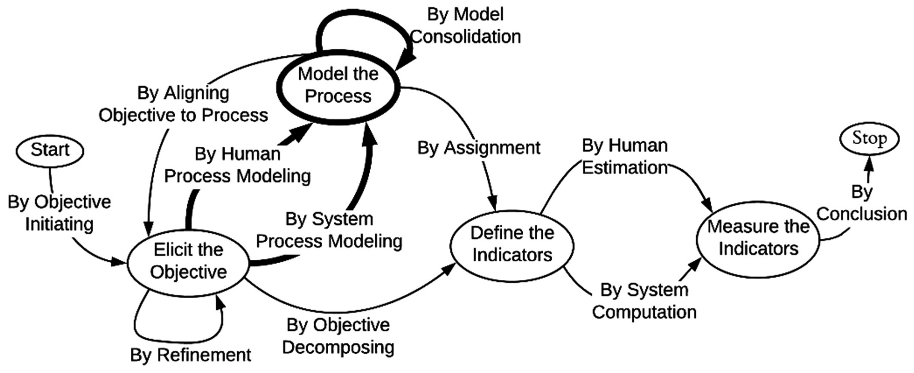


Fig. 1. The intentional map for the process analysis strategy of the CEFOP method [1].

In [1], the process analysis strategy was detailed by identifying four main intentions to be achieved as shown in Fig. 1.

The process model is one of the intentions to be attained during this analysis. This model is used for: the communication between the participants, detection of problems and solutions, prediction and simulation of process evolution. Taking into consideration these usages, the process model must be:

– **Comprehensive:** It should be easy to read and understand by the actors of the process and the process owner, respecting so the pragmatic quality criteria in [3].
– **Complete:** All the activities done by the actors during the process must be represented in the model.
– **Aligned to the reality:** It should describe how the process is really executed, illustrating what is happening, as defined in [4].
– **Useful:** The actors and process owner can perform in-depth analysis [4] and set-up indicators based upon their needs.

In this article, we are going over the existing approaches targeting the process modelling intention and introducing the model consolidation strategy. This strategy aims to merge the models derived by business process modelling and process mining techniques into one unique consolidated model that contains two levels of abstractions:

– The low-level abstraction illustrating the actor' activity at a fine-grained degree.
– The high-level abstraction describing the global activities of the process performed by the actors.

The steps proposed for consolidating the model are described and illustrated by using a real-life process provided by our collaborator, Net Invaders. The client management process covers the activities from the client inscription up to the purchase and use of the solution proposed by the company. The triggering activity is the client inscription. This request is validated by the client manager by a first telephonic contact and after that the client's testing environment is created. Once the environment is ready, the client may start testing the solution. At the end of the testing period the client can decide to purchase or not the solution. If the solution is purchased, a contract is signed between the parties and the client can start using the solution and get charged for this service. Else the client environment is deleted, by rolling back all the configurations and erasing all the actions performed by the client during the testing period.

In Sect. 2, we are overviewing several existing methods and techniques in disciplines of the process discovery and business process model and illustrating the models of the process obtained for the case study. Section 3 details the steps to be followed for consolidating the model and mapping the two levels of abstractions. These steps are illustrated with the results perceived during the model consolidation in the client management process. We conclude this article and discus over perspectives on Sect. 4.

2 Related Works

Business process modelling and process discovery are the two principal disciplines identified in [3] that aim modelling the business process. These disciplines follow two distinct strategies for attaining the intention of process modelling:

The business process modelling discipline follows a human process modelling strategy. A team of persons composed of business analysts, actors of the process or both is charged to model the process.

The automate process discovery, on the other hand, follows a strategy of process modelling by system. The model of the process is automatically discovered by applying process mining techniques over the traces left during the process execution.

Several methods and techniques are proposed by both disciplines for modelling a process based on the context. In this section we are going over some of the most relevant methods proposed by each of these disciplines and illustrating the models obtained by them for the client management process.

2.1 Business Process Modelling

The business process modelling discipline proposes different methods and techniques for attaining the process model intention. These methods and techniques are grouped into two principal categories based upon the participation of the actors of the process into the modelling: participatory and non-participatory. We are focusing principally on the participative methods, the actor's participation into the process modelling reduces the intervention of an external expert and increases their understanding of the model. This last element reduces the internal resistance when it comes to diagnosing and evolving the process.

The intervention of the actors of the process into the modelling process generate a more comprehensive and complete model of the process as stated in [5]. In order to facilitate their participation and motivation, the concept of serious games was used in several methods. In [7], virtual modelling technique introduced the usage of avatars in the 3D environment to model the process. A similar technique was used also in ImProve presented in [8], for helping the actor of the process to better understand and improve their process. These methods provide a visual interface so that the actor can better interact with the model but a pre-training is required for taking into hand the modelling process. The usage of tangible artifacts lowers this entrance barrier as presented in CoDesign [10]. CoDesign and ISEA [9] propose the usage of posted and other tangible tools to model the process. The actors are encouraged to exchange between them and better understand the model. ISEA provides a more structured guide to be followed when modelling the process and a mean of converting the model into a formal format. The actors of the process are guided in modelling the process themselves, by taking turn and describing their activities and the documents used during these activities.

We used ISEA method to model the process of the client management. The actors described their activities and the used tools instead of the documents. The model generated is shown in Fig. 2.

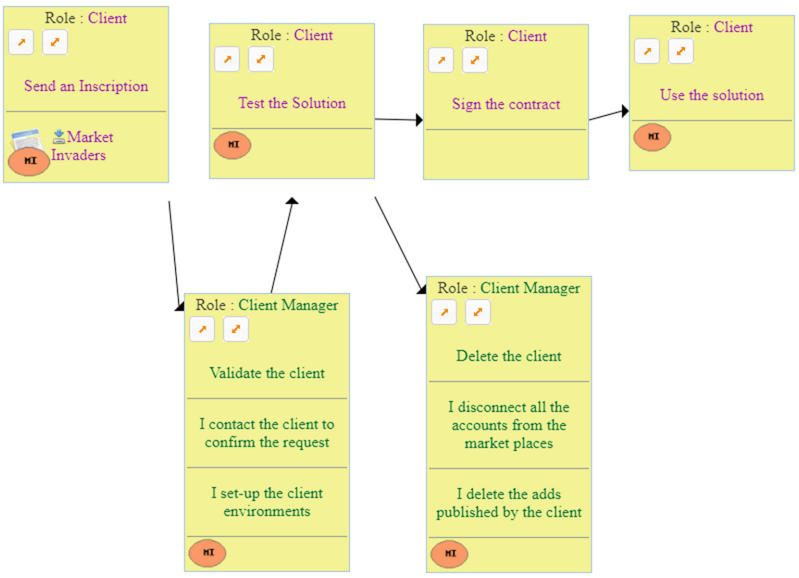


Fig. 2. The process of the client management modelled by its actors using the ISEA method.

The obtained model is quite comprehensive: It is easy to read and understand the principal activities and the paths between them. The actors were able to describe all the activities that they performed assuring so the model completeness.

This model however is descriptive, the actors outlined the process and it is not sure that all the paths taken during the process execution are illustrated. When modelling the process, the actors indeed describe the general paths, neglecting possible deviations in the model. Another disadvantage of this model is that it doesn't offer a lower level of abstraction where it is possible to analyse the actions performed by the actors within their activities. Also no mean of quantifying the process performance is provided since there is no correlation between the models and the data upon which the indicators could be calculated.

2.2 Process Mining

Several techniques and methods are proposed by the process mining discipline. One of the principal method of this discipline is PM2 [11]. PM2 is composed of seven steps, guiding from data collection up to the identification of the problems in the current state of the process. The model discovered from process mining is based upon the event logs that are generated during the process execution, thus it is aligned to the reality. The usage of event logs to model a process allow to perform a process analysis at the action level and to set-up and compute indicators based on the traces of the process execution. However, the dependence from the event logs, limits process mining techniques in the discovery of activities that are not reflected in the event logs, such as manual activities performed in the process.

Moreover, real-life processes generate complex event logs, containing large number of events on a very fine-grained state and multiple paths. The models discovered in such case are complex and referred to as spaghetti models [15]. To discover a model for our case study, an event logs file containing 70 event names [14] and around two hundred cases was extracted from the tool used during this process. We use the DISCO [13] process mining tool to discover the model shown in Fig. 3.

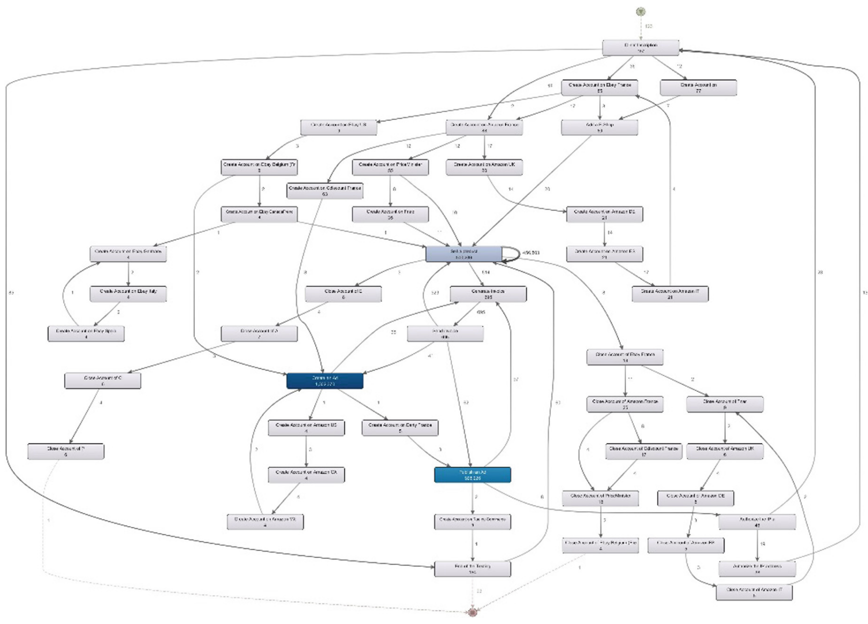


Fig. 3. The model of the client management process discovered by using DISCO.

The discovered model is not comprehensive and no analyse of the process can be performed upon it. Several techniques are proposed for structuring these spaghetti models. L* method introduced in [12] suggests to decompose such processes into sub-processes that are easier to understand and analyse. The disadvantages of such solution are the impossibilities to have a global view over the process and to choose the right way of decomposing the process that fits the focus of the analyse.

Most recent techniques in process mining permit to deal with spaghetti models by creating a higher abstract level by using supervised [16] and unsupervised [17] learning techniques. These learning techniques are applied over the event logs file in order to create a higher level of abstraction within the events. It is upon the new created level of abstraction of the event logs that a more structured model is discovered. In [16], the abstraction of the events is performed by using a training dataset. This dataset is a set of events already labelled by a domain specialist as belonging to a given activity on the higher level of abstraction. It is up to the supervised learning technique to assign the rest of the events on these activities based on their similarities to the trained dataset. The disadvantage of this technique is that a training dataset has to be prepared by a domain expert and the activities belonging to the high levels must be already defined.

In [17], on the other hand, the abstraction of the events is done by an unsupervised technique. The events are grouped based upon patterns of executions. Once the abstraction is performed, it is up to the domain specialist to properly name the high-level events. In both cases on [16, 17], the intervention of a domain expert is required for completing the higher level of abstraction and the actors of the process are left aside.

2.3 Overview of the State of Art and Positioning

Two main strategies are followed in the existing state of the art for attaining the intention of the process model: the human and system process modelling. Figures 2 and 3 show the models of the client management process derived by these strategies.

The ISEA method was used in the case of the human process modelling strategy since it puts the actors of the process in the centre of the modelling process and offers a full guide for converting this model into a formal format. The actors are gather around the table and charge to model the process by exchanging between them. The derived model is comprehensive and complete, but it lacks alignment to the reality, uses for performing in depth analysis and means of measuring the performance of the process.

We used the fuzzy miner to discover a model of the process when undertaking the strategy of system process modelling. The model derived by this approach is aligned to the reality but not complete. The activity of *"Sign the contract"* is not represent in this model since it is a manual activity and it produces no trace in the system. Another disadvantage is that this model is uncomprehensive and cannot be used for analysis. Even though the possibility of creating a higher level of abstraction in the model is proposed, the mapping of the transition between the two levels of abstraction requires the intervention of an external expert and leaves aside the actors of the process.

In order to have a model of the process that complies to the four criteria and fill the gaps in the state of the art summarized in Table 1, we are introducing in the following section the model consolidation strategy.

This strategy aims to merge the models derived by the two existing approaches into one unique model. Since these models are used as starting points for the model consolidation, the model consolidation strategy inherits the context's requirement of:

Table 1. The overview of the state of the art illustrating the existing gaps in the related works.

	Criteria				
	Complete	Comprehensive	Aligned to reality	Useful	
				Analysis	Indicators
Model of the process ISEA	Yes	Yes		High-level	No
Model of the process Heuristic Miner	No	No	Yes	Low-level	Yes

The existence of event logs: This element is a restriction for undertaking the system modelling strategy, since the model is discovered based on the traces left during the process execution.

The actor's participation: This restriction is an essential criterion for assuring the continual evolution of the process in the CEFOP method, since it limits the intervention of an external expert and the internal resistance in the process evolution.

3 Process Model Consolidation

We are enriching the current state of the art, with the model consolidation strategy as show in Fig. 1. The aim of this strategy is to merge the models derived by the human and system process modelling strategies into one unique model, by structuring them as two different levels of abstractions for the same process model, where:

– The low-level abstraction is derived by system modelling strategy. The model illustrates the activities of the actors at a fine-grained degree, by illustrating the performed actions.
– The high-level abstraction is derived by the human modelling strategy, showing the global activities performed by the actors.

The model consolidation strategy guides the actor of the process into creating a map for travelling from the low-level to the high-level of model abstraction, allowing so the mutual enrichment between the models. This strategy permits to attain a model of the process that is comprehensive, complete, aligned to the reality and useful for analysis and process performance measuring. The model consolidation strategy is carried out in four steps. The tasks performed in each of these steps are detailed in the following subsection and illustrated on the client management process. We are using the terms:

- The process model to refer to the BPMN format of the described model of the process (Fig. 2), automatically retrieve by ISEA. This model is printed on a big format such that action cards can be placed within the activities. The process model of the client management is shown in Fig. 4.
- Activity to refer to an activity described by the actor when modelling the process by human. In the process model, the activities are represented as coloured rectangle,

where the colour defines the role performing it. Each role is represented by a unique colour into the model.

- Role to refer to an actor or a group of actors having the same role in the process. Each role participating in the process must be represented by at least one actor during the model consolidation.
- Action to refer to an event name [14], an activity found in the dataset used for modelling the process by system. The actions are represented by action card during the experiment as illustrated in Fig. 5. The action card can be coloured (with the role corresponding colour) or white (if the role is undefined in the dataset).

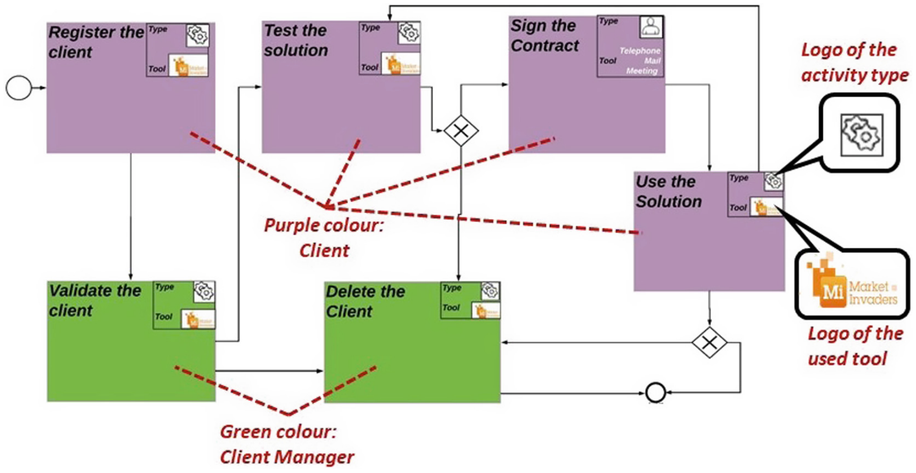


Fig. 4. The process model projected from the model describe by the actors in Fig. 2. The green and purple color represent respectively the role of the client manager and the client. (Color figure online)

3.1 Step 1: Correlate Activities and Actions

This is the first step to be performed when undertaking the model consolidation strategy. In this step, the roles have to associate their actions with the activities they have described. Table 2 describes the tasks to be performed in this step.

When undertaking this step in the process of the client management, the set of 70 action cards were distributed to the participants. Once the roles associated the action cards with the activities, two action cards were places aside one of the existing activity, creating so a new activity as illustrated in Fig. 5.

3.2 Step 2: Enriching the Set of Actions

Once the actions are correlated to the activities, the second step of the model consolidation can take place. The enriching of the set of actions is done in two tasks as detailed in Table 3. The time for the tool analyzing task differs depending on the time required for: verifying if the proposed tool stores traces of the process execution,

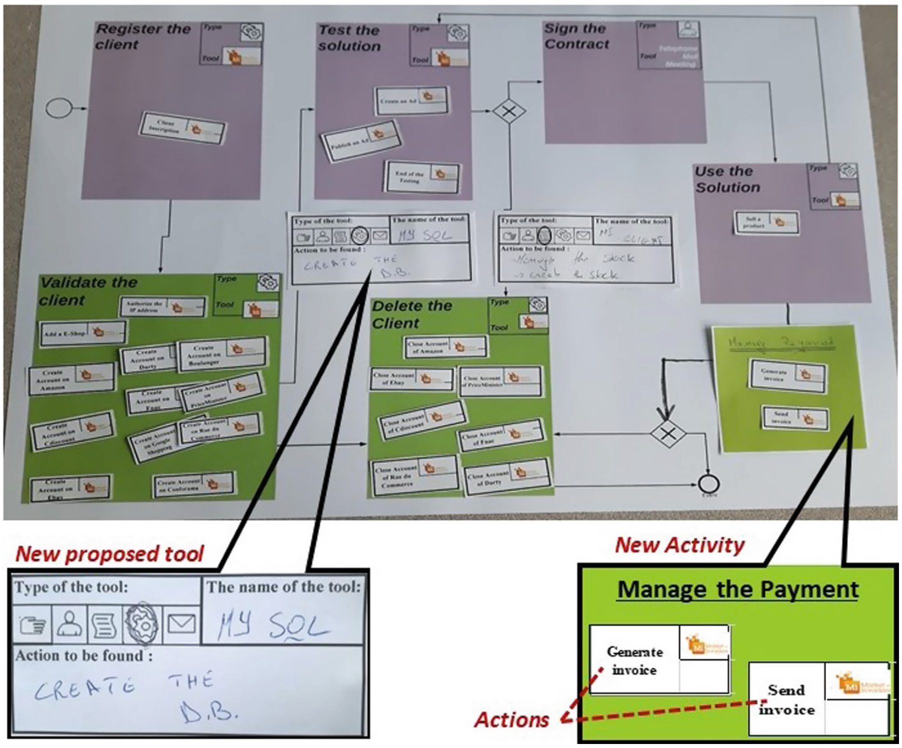


Fig. 5. The figure illustrates the result perceived during the two first steps of the model consolidation on the client management process.

extracting the traces and adapting them to the existing event logs. In case when a new set of traces is extracted by the proposed tool, the correlation of activities with actions must be retaken.

Two new tools were introduced by the actors during the experimentation on the case study as shown in Fig. 4. These sources were analyzed and no traces were able to be extracted since the tools didn't store traces of the process execution. However, a procedure for enabling the generation of logs in both tools is planned for enriching the model in the future.

3.3 Step 3: Paths Consolidation

The goal of this step is to identify deviations between the paths discovered by the system process modelling strategy and the paths described by the human process modelling strategy. The deviations between the paths are identified by the animator and validated by the actors of the process by enriching both models mutually. These tasks are described in Table 4.

In the client management process, three deviations were identified and only two of them were validated by the actors as shown in Fig. 6. The deviations were due to:

Table 2. The tasks performed for correlating the activities with the actions: the supports and the duration previewed for each of them.

Task and Instructions	C/I*	Supports and duration
Preparation: The animator brings the process model and describe the color code to the participants. The roles are asked to: *"Can you describe your activities in 2–3 sentences by following the order of the activities' execution within the process?"*	C	The process Model 10 min
The animator places the action card on the table and roles have to: - If the roles are defined: *"Take the action cards which have the color that was assigned to you"* - If the roles are not defined: *"Go over the action cards and pick the one's that represent actions that your perform"*		The action cards 5 min
The roles are asked to: "*Distribute the action cards over the process model"* ***Attention:*** *The action cards can be placed only:* *– inside the activity if you do the action within an activity,* *– over the input/output arrows if the action is not performed within an existing activity, but it is performed before/after it*		10 min
The animator asks the roles: "*Please give me the action cards that you have not placed in the process model?"* For each left aside action card, the animator filters the cases containing the action and discovers the model for this set of cases using Pro-M [18]. The simplified model is used to narrates the path of the action execution by describing the actions performed before and after. Once the path is described, the roles are asked: *"Do you want to place this action card into the model or discard it definitively*?"	C	The discovered process model 3 min/action card
The roles are equipped with markers and empty activities cards on their corresponding color and asked to: "*You can use the empty activity cards to create new activities in the model in order to group the action cards left outside the existing activities, for this:* **1.** *Go over the action cards placed on each side of the existing activities and see if they can be grouped at the same activity* **2.** *Once the grouping is done, place an empty activity card under the action cards and title this new activity* **3.** *Using the marker, outline the paths between the new and the exiting activities*	I	Empty activity card Black markers 10 min

*Collective/Individual

The first validated deviation, from *"Test the solution"* to *"Use the solution"* is caused by the absence of the actions reflecting the manual activity *"Sign the contract"* performed in between.

Table 3. Describes the tasks to be performed for enriching the set of actions.

Task and Instructions	C/I*	Supports and duration
The animator distributes tool cards to the roles and asks them: *Are there actions that you performed in this process that are not yet represented in the process model?* *If yes, use a tool card to describe the tool that you use when performing these actions?*	C	Tool cards 10 min
The animator goes over the tool cards proposed by the role and 1. Identify the new tools to be analyzed, by filtering the tools types (non-manual) that can be further inspected for traces 2. Extract the traces from the tool if possible	I	The required time is variable

*Collective/Individual

Table 4. The tasks performed for identifying deviations between the paths and validating them.

Task and Instructions	C/I*	Supports and duration
The animator is charged to detect deviations between the paths in the two models. The deviations are detected by following the steps: 1. Identify the paths between actions. All the paths discovered by using Pro-M, heuristic miner algorithm over the dataset are identified 2. Abstract the paths. The paths between actions are abstracted to the higher level, between activities A and B when: It exists at least one discovered path between events a and b, where a is the set of events belonging to activity A and b is the set of event belonging to activity B 3. Find the deviations by comparing the discovered paths with the described paths in the process model	I	Pro-M Dataset of actions 10 min
The deviations are overviewed one by one. The animator, using a red marker, draws the deviation path into the process model and asks the roles: *"Do you know this path or you need to decompose it and inspect it at the actions level?"* Once the path is clear, the actor must decide to validate it or not: *"Which of the following reason better describe this deviation and do you want to keep this path or not?* *(a) The path was forgotten when the model was described* *(b) The path is an exception; taken in exceptional cases* *(c) Actions are missing in the model. An action is performed between the two edges but it is not represented by an action card* *(d) Existing actions must be duplicated (or moved): The edge actions for this path are present in the model, but their position need to be reviewed* If the path is validated, the animator uses a black highlighter to overwrite the red path In cases where the reason of the deviations are (c) or (d), step 1 must be retaken	C	The discovered process model Black and red highlighter The deviations

*Collective/Individual

The validated deviation, from *"Use the solution"* to *"Test the solution"* reflects a deviation that is currently happening and must be corrected in the future.

The non-validated path, shown in red in Fig. 6 represents a deviation previously detected and already rectified in the system, so there is no need to keep it for further analysis.

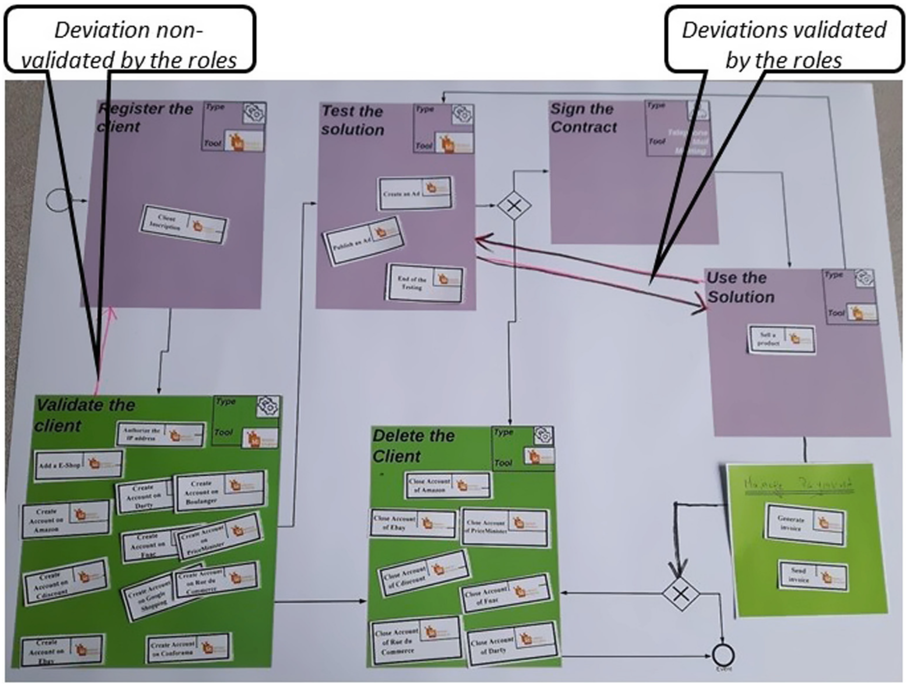


Fig. 6. The result of the path consolidation step in the client management process is illustrated, by showing the validated and the non-validated paths.

3.4 Step 4: Consolidate the Models

This is the last step in the process consolidation. The aim of this step is to mutually consolidate both models, for this, two tasks are performed in parallel:

1. Cleaning the dataset: The aim of this task is to remove from the dataset all the non-used actions and all the cases containing the non-validated paths.
2. Enriching the high-level model: All the validated deviations and new activities emerged during the process consolidation must be inserted into the high-level model that was perceived through the actor's description.

In our case study, the event logs were cleaned by removing all the cases containing the path from "*Validate the client"* to *"Register the client"* and the described model were enriched with the new activity "*Manage payment"* and with the path from *"Use the solution"* to *"Test the solution"*.

4 Conclusions and Perspectives

Two main strategies exist nowadays for modelling a process: human process modelling and system process modelling. Neither of them is able to provide a process model that is complete, comprehensive, aligned to the reality and useful for analysis. In order to attain such objective, we introduced in this article a model consolidation strategy that merge the models derived by these strategies into one unique model with two different levels of abstraction. This consolidation permits to have a model that is:

Comprehensive since the high-level model abstraction is created by the actors of the process and it is understandable and readable by them.

Complete: All the activities performed by the actors are reflected at the high-level of abstraction.

Aligned to the reality since the mapping between the levels of abstraction enriches and correlates both models mutually.

Usable for depth analysis: During the mapping of the two levels of model abstraction a correlation between event logs and activities is settled. This correlation facilitates setting up indicators and their computation.

During the undertaking of model consolidation strategy beside the model's correlation, the actors of the process are also:

Cleaning and enriching the event logs: The identification of non-used actions or paths cleans the event logs from noisy data that might be present in the event log. The consolidation incites them into identifying new sources of traces that can further enrich the model and the analysis.

Apprehend the models: The implication of the actors in the model consolidation, permits them to better understand the models, its deviations and the need of changing the process.

In this article, we illustrated an experimentation of the model consolidation performed over a process having a set of 70 event names. In order to facilitate the undertaking of the model consolidation strategy for other process having hundreds of event names, in the future works we want to provide an interface that facilitate the correlation between the actions and activities. The automatically detect the deviations between the models is also a step to be put in place, since currently this task is performed manually demanding so that the animator must have process mining knowledge.

The process model is one of the four intentions aimed to be achieved when analysing the state of a process as shown in the intentional map in Fig. 1. In our future works, we aim to define how to attain the three remaining intentions: define the objective, define the indicators and measure the indicators.

References

1. Çela, O., Front, A., Rieu, D.: CEFOP: a method for the continual evolution of organisational processes. In: 11th International Conference on Research Challenges in Information Science (RCIS) (2017)
2. Front, A., Rieu, D., Cela, O., Movahedian, F.: Les méthodes d'évolution continue au sein des organisations: le cadre As-Is/As-If. In: INFORSID (2017)
3. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management, 1st edn. Springer, Berlin (2013). https://doi.org/10.1007/978-3-642-33143-5
4. Van der Aalst, W.M.P.: What makes a good process model? Softw. Syst. Model **11**(4), 557–569 (2012)
5. Rittgen, P.: End-user involvement and team factors in business process modeling. In: 45th Hawaii International Conference on System Sciences, Hawai, USA (2012)
6. Buur, J., Ankenbrand, B., Mitchell, R.: Participatory business modelling. CoDesign: Int. J. CoCreation Des. Arts **9**(1), 55–71 (2013)
7. Brown, R., Recker, J., West, S.: Using virtual worlds for collaborative business process modeling. Bus. Process Manag. J. **17**(3), 546–564 (2010)
8. Ribeiro, C., Fernandes, J., Lourenço, A., Borbinha, J., Pereira, J.: Using serious games to teach business process modelling and simulation. In: International Conference of Modelling, Simulation and Visualisation Methods, Las Vegas, USA (2012)
9. Front, A., Rieu, D., Santorum, M., Movahedian, F.: A participative end-user method for multi-perspective business process elicitation and improvement. Softw. Syst. Model., 1–24 (2017)
10. Herzberg, N., Kunze, M.: The business process game. In: 7th Central-European Workshop on Services and their Composition (2015)
11. van Eck, M.L., Lu, X., Leemans, S.J.J., van der Aalst, W.M.P.: PM^2: a process mining project methodology. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_19
12. Van der Aalst, W.M.P.: Process mining: discovering and improving Spaghetti and Lasagna processes. In: IEEE Symposium on Computational Intelligence and Data Mining (2011)
13. DISCO tool Homepage. https://fluxicon.com/products/
14. van der Aalst, W.M.P.: What kind of data does process mining require? http://www.processmining.org/logs/start
15. Van der Aalst, W.M.P.: Discovery, Conformance and Enhancement of Business Processes, 1st edn. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-19345-3
16. Tax, N., Sidorova, N., Haakma, R., van der Aalst, W.M.P.: Event abstraction for process mining using supervised learning techniques. In: Bi, Y., Kapoor, S., Bhatia, R. (eds.) IntelliSys 2016. LNNS, vol. 15, pp. 251–269. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-56994-9_18
17. Mannhardt, F., Tax, N., Unsupervised event abstraction using pattern abstraction and local process models. In: Enabling Business Transformation by Business Process Modeling, Development, and Support Working Conference-BPMDS (2017)
18. Pro-M tool Homepage. http://www.promtools.org/doku.php
19. Net Invaders company homepage. https://en.net-invaders.com/

From Instance Spanning Models to Instance Spanning Rules

Manuel Gall(✉) and Stefanie Rinderle-Ma

Faculty of Computer Science, University of Vienna, Vienna, Austria
{manuel.gall,stefanie.rinderle-ma}@univie.ac.at

Abstract. Instance Spanning Constraints (ISCs) allow to control the behavior of multiple business processes and their instances which is crucial in many application domains (e.g., for process synchronization). The modeling and visualization of ISCs hence constitutes an essential brick in process compliance management. Currently, no approach exists for representing a set of Instance Spanning Rules (ISRs) based on an ISC. Existing work rather focuses on visualizing ISCs at the more abstract Instance Spanning Model (ISM) level. However, the gap between the ISM and the ISR level must be bridged in order to enable the enactment of an ISC at process level. Hence, this paper collects requirements for the implementation of ISRs through an ISM, e.g., by specification of data elements. Based on the requirements an existing visual modeling language is tailored towards the modeling of ISRs and the corresponding XML specifications are provided. Both, visual modeling and XML representation are prototypically implemented and illustrated by means of a set of use cases. Finally, an algorithm for deriving the common ISM of a set of ISRs is introduced and evaluated based on a given test set.

Keywords: Instance Spanning Constraints
Instance Spanning Rules · Compliance · Rule visualization
Instantiable constraint · Process-aware information systems

1 Introduction

Business processes compliance deals with the enforcement of constraints such as new laws and regulations. Over time business process compliance became a billion dollar market [6]. Nowadays compliance rules tend to incorporate more and more business knowledge. Recent studies show that constraints spanning multiple instances and or processes are omnipresent [4]. Some approaches to deal with instance spanning constraints (ISCs) exist, e.g., [4,7,14]. However, the development life cycle from extracting, modeling, and implementing ISCs is not fully supported yet.

Contrary, for intra instance compliance the development life cycle has been established and starts with a constraint definition, e.g., a new law or regulation, followed by creating a non-executable model mainly for communication purposes [3]. This model is refined into instantiable rules which can be checked for process instances [11].

Establishing the development life cycle for ISCs requires a visual modeling notation in order to create instance spanning models (ISM). For this, for example, the visual notation ISC_Viz [5] can be used. However, ISMs are not executable. Hence, an ISM has to be instantiated and specified into executable instance spanning rules (ISRs), but an approach for this is missing.

This work aims at bridging the gap between ISM and ISRs and vice versa based on the following questions: "(1) How to go from an ISM towards a set of instance spanning rules (ISRs) – visually and at the implementation level?" and "(2) How to merge a set of ISRs into an ISM again?". The second question is important particularly in large, living systems where for a given set of ISRs the common schema would be important for validation. In order to answer these questions we use design science research [17]. A set of requirements is gathered from literature. Based on these requirements we build the following artifacts: a visual notation for ISRs, a corresponding XML representation, and a merge algorithm. We evaluate the applicability on a set of real world examples and compare the merge results against previous research.

The paper is structured as follows: Sect. 2 provides a motivating example. Section 3 discusses related approaches. Requirements are collected in Sect. 4. Our proposed solution for ISRs is discussed in Sect. 5. Section 6 introduces a merge algorithm for ISRs. The applicability of our approach is discussed in Sects. 7 and 8 concludes with a summary.

2 Motivating Example

We lay out the basic concepts of the constraint development life cycle and management by means of an example. A vial has to be examined. Figure 1Ⓐ shows the process in BPMN notation. The process consists of three activities *PreTest*, *Centrifugation* and *PostTest*. In the *PreTest* the vial is examined. In the *Centrifugation* activity the vial is put into the centrifuge and the centrifuge is started and stopped when centrifugation is finished. Afterwards the vial is taken out of the centrifuge. In the last activity a post examination is performed.

Figure 1Ⓑ shows an example for an intra-instance instantiable rule modeled with BPMN-Q [2]. It states that the *PreTest* has to precede the *PostTest*. Furthermore the figure shows two instances 1 and 2. Instance 1 complies to the constraint while 2 violates it. The decision on compliance or violation can be made for each instance in a separate manner. What we cannot see in the visualization is that in order to perform such a compliance check each task has to be linked to a task within the process.

Figure 1Ⓓ depicts all modeling elements from the ISM language used in this paper. *Dataelements* and *Resources* represent values specified during design or run-time and either within the rule or the process model. A *Conditional - Trigger* uses *Dataelements* and *Resources* to validate a business rule. The *Timer - Trigger* extends a *Conditional* with time related information. If a business rule

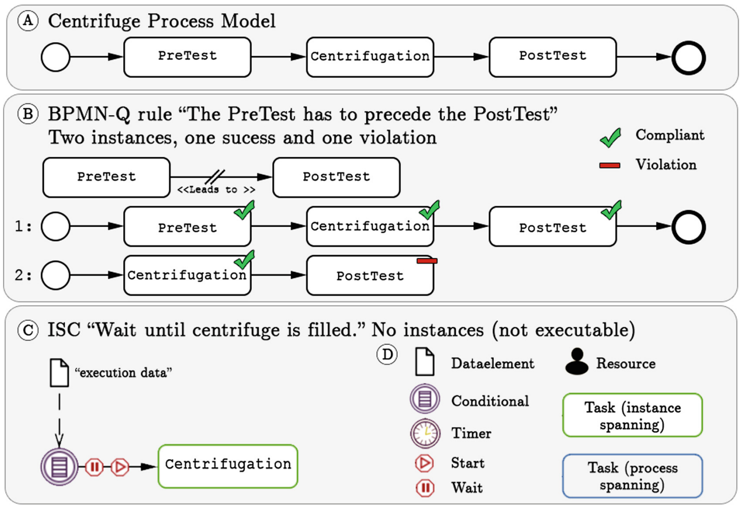


Fig. 1. Centrifugation process with intra-instance constraints and ISM. (Color figure online)

within a trigger is true the action part is executed, i.e., *Start* the execution of a task or *Wait* before a task is executed. Instance spanning tasks are represented with a green border while process spanning tasks with a blue one.

Figure 1 Ⓒ depicts an example ISC "Wait until centrifuge is filled." [15]. What differentiates an ISC from an intra-instance constraint is that the ISC involves multiple instances and/or processes. In this example the centrifuge offers a maximum load capacity of 4 vials and shall only be started when the maximum load capacity is reached. We used ISC_Viz to model the corresponding ISM. The visualization shows a *trigger* which performs a check resulting in a boolean value. A *data element* that stores a numeric value. Two *actions* wait and start and a *task*. What we cannot see is that the task is currently not linked to a process or instance. Based on the color *green* we know that the task has to be linked to one process in order to be executable. In its current state this model is not executable, the trigger does not state a condition, the data element and task are represented only by a label and the execution order of the actions is unclear.

3 Related Work

Most studies on constraints focus on modeling intra-instance constraints [3] or instantiating constraints [6,10,12] rather than widening the view towards ISCs. However, recent approaches such as [4,7,8] concentrate more and more on constraints spanning across instances and processes. Batching of multiple instances [14] is a common scenario when dealing with ISCs, however the work focuses on integrating batching regions into the process rather then separating the logic. At a formal level recent work even deals with going towards instantiation [4,8,9,16].

Also first steps are taken towards the discovery of ISCs from process logs [18]. Current approaches for visualizing constraints focus on intra-instance constraints [1,11]. These approaches can not be used when dealing with spanning constraints due to more data and the involvement of multiple instances and or processes. ISC_Viz [5] is a visual notion for modeling ISMs. However, the language is limited to visually represent models for communication purpose only. The created models are not instantiable. With our research we want to extend ISC_Viz towards the modeling of ISRs.

4 Requirements for ISR Representation

This section states the requirements for the specification of ISRs based on an ISM. The requirements are derived from literature on instantiation of intra-instance compliance rules [11], instance-spanning constraints in general [4], and the visualization of ISMs [5]. This selection of approaches is feasible as the classification and visualization of instance-spanning constraints yields the necessary ingredients to go from their abstract and process-independent representation to an executable form. The related work on intra-instance constraints confirms these specification steps going from process-independent to process-specific rules. However, as ISCs contain additional, runtime related information such as trigger and actions when compared to intra-instance constraints, the approach in [11] cannot be directly applied to ISCs.

Nonetheless, when specifying executable rules the runtime-related parts of ISCs have to be specified, i.e., the data, the resources, and the linkage, leading to the following set of requirements:

Requirements Derived from Intra-instance Constraints

1. *Data specification:* When an ISM is transformed into a rule all data elements have to be specified. For example, an ISM states that a loan amount is needed. In case of this example a rule either assigns a specific value to the variable or links to a source where to read the variables value.
2. *Resource specification:* While an ISM gives an overview of involved resources, a rule has to have a link to the resource and allow for e.g., checking if the resource is available.
3. *Linkage:* An ISM only states activity names. A rule has to specify a link between each activity of the rule and an activity within a process. An exception of this behavior are activities that link to no specific process, in this case the activity links to a repository of activities. An example rule that demonstrates this behaviour can be found in Sect. 7.

As mentioned before instance-spanning constraints contain additional information when compared to intra-instance constraints [4] which is necessary to define their scope (spanning instances or processes), their trigger (when do they become activated), and their actions, e.g., waiting.

Requirements Derived from Instance-spanning Constraints

4. *Execution data*: Requirement 1 already handles that data elements are to be specified before a rule can be created. Execution data needs special treatment as this data is generated during process execution and thus is a specification of ISMs and not part of an intra-instance constraint.
5. *Trigger specification:* In order to allow validation of both trigger types *Conditional* and *Timer* a condition and behaviour have to be specified. A condition for the *Conditional* is specified by using data and resources in order to create a boolean condition, e.g., instance counter == load_capacity. A *Timer* condition allows for modeling the same condition as a *Conditional* with the addition of a time event, e.g., lastExecution >1 h && instance_counter == load_capacity. The behaviour of a trigger is executed only if the condition is evaluated as true and enables modeling of data and resource manipulation.
6. *Action specification:* Some actions need to be specified in more detail when handling an ISR e.g. an *Alert* consists of a message, video, sound, or service that shall be executed.

An additional goal of this paper is to automatically create an ISM based on ISRs. This necessitates the following technical requirements.

Requirements for Automatic ISR Creation

7. *Unique ISM identifier:* Every set of ISRs has to store an ISM ID in order to enable grouping of ISRs e.g. a set of 3 ISRs stores the ISM ID of its corresponding ISM. If there is no corresponding ISM a unique value is created.
8. *Unique ISR identifier:* Every ISR is identified by an unique ID.

Current approaches for visualizing an ISM will be used to visualize ISRs. To allow the user to understand both, ISM and ISRs, we will follow a set of strict requirements on the visualization. Those requirements are based on Moodys principles [13] and ensure that users who are familiar with ISM modeling will have no problem in understanding and modeling ISRs.

Requirements on ISR Visualization

9. *Principle of graphic economy:* The complexity of the constraint visualization shall not be changed. Therefore it is not allowed to add additional elements to the visualization, e.g., new symbols.
10. *Semiotic Clarity:* Avoid the definition of multiple semantics for a visual element.

5 ISR Representations

This section provides a visual as well as an XML-based representation for ISRs following the requirements as set out in Sect. 4. As ISRs are typically specified based on an ISM the visual ISR representation has to be designed in accordance with the visual ISM representation. ISC_Viz [5] is currently the only visual modeling language for ISMs. Thus ISC_Viz will serve as basis for the visual ISR modeling language. It will be discussed which of the ISC-related information is visualized, and which is only contained within the xml representation.

5.1 Visual ISR Representation

Our focus for developing a visual ISR representation are Requirements 9 and 10 as stated in Sect. 4: 9 – no further visual elements when compared to ISC_Viz – and 10 – avoid multiple semantics per visual element. In order to meet these requirements we use the principle of complexity management and the principle of dual coding as proposed by Moody [13].

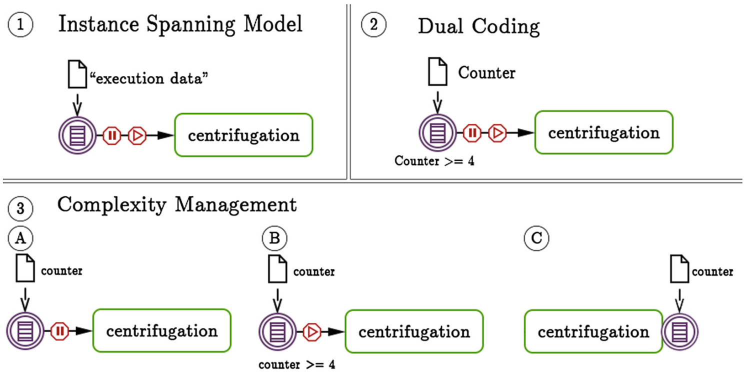


Fig. 2. Representing an ISM as a set of ISRs (example).

Dual Coding. Dual coding uses text in order to complement graphics. We will use this principle to add text to visual elements of an ISM, i.e., data, resource, task, and trigger, in order to meet Requirements 1–4 and partly 5. As text is already used within ISMs this is not considered as adding a new visual element and thus does not violate Requirement 9.

Applying dual coding on the ISM running example of Fig. 2① results in the ISR②. During the creation process, local data element *Counter*, a link to a task within a process as well as a condition and behaviour are specified.

While the specification of ISRs with respect to data, resources, and tasks at the visual level is sufficient using text, at the implementation level, the modeler has to specify either a link to a certain data element, resource and task within a process or repository and/or create a local variable within the rules. To avoid visual overload [13] we decided to not include URLs within the visual notion. Specifying URLs and values allows the rule execution engine access to values used within process instances and thus be used for evaluating conditions. Conditions and their behaviour have to be specified by the modeller. A condition is either true and the behaviour is executed or false and nothing else happens. As far as visualizing the condition and behaviour we opted to visualize the condition of a trigger and specify the details of the behaviour within the XML representation.

Dual coding enables the specification of several ISM elements such as data, but is not yet sufficient to realize executable ISRs. What is missing is the specification of the behavioural parts, i.e., the actions, that are triggered based on the specified conditions. In Fig. 2②, for example, we cannot decide what happens when the *Counter* reaches a value of four or greater. Shall the wait or start action be executed?

In addition, using dual coding only does not enable the specification of different conditions, i.e., the creation of ISRs that possess different conditions, but follow the same ISM. To tackle the specification of actions and multiple conditions, the visual representation of ISRs has to be further refined.

Complexity Management. The ISRs shall include a mechanism to allow for complexity management. Moody suggests two ways of dealing with complexity, modularization and hierarchy. Modularization divides a problem into multiple smaller problems. Hierarchy allows to represent a problem at different levels of detail. We will use this principle to refine the ISR visualization and tackle Requirement 5. We split one ISM into multiple rules. Each rule consists of one trigger and task, and zero to multiple resources, data elements, and actions. This allows the rule execution engine to know what behaviour and what action shall be executed in case a condition is true. A special type of trigger is a trigger without a condition such a trigger is always true.

Figure 2③ depicts the application of complexity management and dual coding on the ISM①. ISRⒶ shows a rule without a condition and action wait. A rule engine will stop all instances before *centrifugation* is executed. The visualization does not show the default value of the *counter* (0) and the behaviour that every time this rule is true the *counter* is increased by 1. ISRⒷ depicts a condition $counter >= 4$ and a start action. This means that if an execution engine evaluates this rule as true the following task (i.e., *centrifugation* in the example) will be started. ISRⒸ does not state a condition but we can see that the *counter* is used again. This rule can be seen as a clean up. Within the behaviour we decrease the counter by one for each instance that finished the *centrifugation* task. If the process engine supports instance counting the *counter* of the engine can be used. In this case increasing and decreasing the counter within the rule engine is not needed.

In comparison to the ISM the ISRs show more information. When looking at the ISRs we know that we always stop the execution of the instances before the task *centrifugation*. We know that we start the instances when we reached the fourth instance. Specifying URLs and/or values for all elements creates a set of executable rules. The following XML representation details information that is not shown within the visual representation, e.g., URLs, behaviour and IDs.

5.2 XML Representation

When transforming an ISM to ISRs there is more information needed than shown in the visualization, i.e., URLs, IDs, and behaviour. We use XML as file format to store all the necessary information for communication with our execution and rule engine as both support XML files as input and output. Furthermore we opted for XML as it allows for validation against a schema. The schema describes all necessary information we know from the visualization as well as some "hidden" information. We will highlight this hidden information through the following discussion. The following code snippets are a short excerpt from the detailed Relax NG schema available here[1].

The visualization of data elements only visualizes their *label*, but to satisfy Requirements 1 and 4, we have to store additional information. Every *data* element consists of a *type*, e.g., *data* and *execution data*, a *label* and either *content* or an *url* and an *id*. Data elements that specify a value, e.g., MaximumNumber of concurrent instances in centrifuge, will most likely use the element *content* and specify the value. A more dynamic approach allows to link a value to the process itself by specifying the element *url* and *id*. Linking a value to a process allows access to process data and the possibility to use this data for *trigger conditions* and *behaviour*.

Listing 1.1. Relax NG data elements representation

```
<define name="dataelements">
  <element name="dataelements">
    <zeroOrMore>
      <element name="data">
        <element name="type"><choice>
          <value>data</value><value>execution</value>
        </choice></element>
        <element name="label"><text/></element>
        <choice>
          <element name="content"><text/></element>
          <group><element name="url"><text/></element><element name
              ="id"><text/></element></group>
<choice>  </element>  </zeroOrMore>  </element>  </define>
```

From a visual and XML perspective *resources* and *data* are nearly handled the same. As stated in Requirement 2 – a resource always links to a process or repository – we have to represent this behaviour within the XML representation. To do so we use an *URL* and *ID* pointing to a resource. The idea is that every resource is represented with a web interface i.e., all printer "/printer/id" are represented as rest services and to get data from this device one uses the URL and ID to for example determine the current ink level "/printer/id/inklevel".

We assume that most of the instantiable rules consist of one *action*. But there might be cases where multiple actions are possible, e.g., execute an alert and then perform the *action* wait. To satisfy Requirement 6 an optional URL can be used for user notifications. An alert can show a message to a user, but one can think of further use cases like playing a sound or opening a service.

The *trigger* visualization is a bit different compared to other visualizations. A *trigger* can be represented in two ways by a *conditional* or a *timer*. This representation depends on what is stated within the XML element *type* shown in Listing 1.2. A *type* is specified either as *conditional* or *timer*. While the *condition* is visualized a trigger stores a behavioural part that is not visualized, but needed to meet Requirement 5. Additionally all previous defined elements *data*, *resources* and *actions* are included in the *trigger*.

[1] http://gruppe.wst.univie.ac.at/projects/crisp/index.php?t=visualization.

Listing 1.2. Relax NG trigger representation

```
<define name="trigger">
  <optional>
    <element name="trigger">
      <element name="type"><choice>
        <value>conditional</value><value>timer</value>
      </choice></element>
      <ref name="dataelements"/>
      <ref name="resources"/>
      <element name="condition"><text/></element>
      <element name="behaviour"><text/></element>
      <ref name="actions"/>
    </element>  </optional> </define>
```

Listings 1.1–1.2 are combined within one rule definition shown in Listing 1.3. As specified in Requirement 7 and 8 every rule consists of an ID that groups all rules under a single ID. Additionally we allow for naming and describing a set of rules. Every rule set consists of one or more rules. To allow for identifying a single rule we again use a unique id for every rule within a set of rules. Furthermore rules consist of elements for name, description, and *priority*. The *priority* is needed within the rule engine to determine in which order rules shall be evaluated. A *trigger* before the task signals that the trigger is evaluated before the execution of a task while a *trigger* after a task is evaluated afterwards. As Requirement 3 states that a task within a rule has to be linked to a either a process or a repository. Within the XML file this linking is expressed by an element *spanning* with two values *single* for an URL to a process and *multi* for an URL to a repository. The *url* element specifies under which link the corresponding *process* or *repository* can be found. Within the *process* or *repository* an ID is used to narrow it down to a specific activity. One could argue that we do not store a label within the task, but the visualization of a rule shows a label within the task. To be consistent between the linked *process* or *repository* and the rule we use the label from the linkage. In some cases a task might not be needed, e.g., cleaning variables as shown in Sect. 7, Table 1 ①.

The key differences between the XML and visual representation are that *IDs*, *URLs* and *priority* and some element specifics, e.g., *types*, *behaviour* are not visualized, but stated within the XML file. To the contrary the *label* of a *task* is visualized but it is only indirectly stored within the XML file by specifying a *url* and an *id*.

Listing 1.3. Relax NG combination of all parts

```
<element name="rules">
  <element name="id"><text/></element>
  <element name="name"><text/></element>
  <element name="description"><text/></element>
  <oneOrMore>
  <element name="rule">
    <element name="id"><text/></element>
    <element name="name"><text/></element>
    <element name="description"><text/></element>
    <element name="priority"><data type="integer"/></element>
    <ref name="trigger"/>
    <element name="task">
      <optional>
        <element name="spanning"><choice>
          <value>single</value><value>multi</value>
        </choice></element>
        <element name="url"><text/></element>
        <element name="activityid"><text/></element>
      </optional>  </element>
    <ref name="trigger"/>
  </element>  </oneOrMore>  </element>
```

6 Automatic Creation of an ISM from a Set of ISRs

The specification of ISRs based on an ISM is necessary in order to create executable rules. In addition, the other way round – deriving and ISM from a set of ISRs – is also of interest. Reasons include displaying common elements and structure as well as providing a schema for further ISRs. Creating the ISM from a set of ISRs should be done automatically in order to not burden the user with this task. A precondition for the automatic creation of an ISM from an ISR is that the ISR is valid with the schema set out in Sect. 5.2[2]. It should be possible to select ISRs from a given set for ISM creation. The reason is that the user might not be interested in a fully detailed ISM. This is achieved by selecting a priority value. Only rules with an higher priority will be merged to an ISM.

Algorithm 1 merges a given set of ISRs into an ISM by sorting all rules based on their priority. Then every rule that has a higher priority than a given value is considered for merging. All rule elements such as trigger, data elements, and tasks that are not represented within the generated ISM yet, will be added and simplified. Simplifying means to omit information that an ISM does not contain, i.e., links, conditions, and behavior. The implementation of Algorithm 1 as well as an illustrating example are provided as part of the evaluation in Sect. 7.2.

[2] ISM schema: http://gruppe.wst.univie.ac.at/projects/crisp/index.php?t=visualization.

Algorithm 1. Merging ISRs to create an ISM.

```
$rules = all\ ISRs\ from\ the\ XML\ input$
$priority = value\ from\ input$
$model = null$
$rules.sort()$
while $rules$ do
    if $rules[i].priority > priority$ then
        while $each\ element\ from\ rules[i]$ do
            if $!model.contains(rules[i].element[k])$ then
                $model.add(rules[i].element[k])$
            end if
        end while
    end if
end while
$model = model.simplify()$
```

7 Evaluation

Roughly, the evaluation is conducted in a circle where we start with an ISC and an ISM. We visually specify ISRs based on the ISM. This visual specification of ISRs has been prototypically realized in the modeling tool *ISR modeler*[3] based on the principles introduced in Sect. 5.1. We export and validate these rules with the schema described in Sect. 5.2. Finally we apply Algorithm 1 from Sect. 6 and create an ISM again. We visualize the created ISM with the ISC_Viz[4] modeling tool and compare the visual and XML representation with the original model.

7.1 Creating ISR Visual Models and XML Representations

To show the applicability of the approach, four ISMs are visually specified as ISRs, one for each of the categories of the classification introduced by [4]. In a nutshell, the classification has two dimensions, i.e., requirement and context. Requirement comprises categories single (only one modeling perspective) and multi (multiple perspectives, e.g., data and time) the ISC can be built of. Dimension context refers to single if the ISC is defined for multiple instances of one processes and multi for ISCs spanning multiple processes. The examples depicted in Table 1 are picked from a set of ISC examples [15]. Our visual language is complete in a sense that we can model all 114 of these examples with our language. The corresponding ISMs have been created in our previous work [5]. These ISMs will help to verify the results of merging the created ISRs with Algorithm 1. One has to bear in mind that multiple interpretations of the text are possible as the description is not precise. However Table 1 also comprises specification details of the examples to give more details on how we interpret the rule and what is special about this rule.

Figure 3 shows the visualization of the example ISRs. In the following we discuss how the visualization realizes the requirements set out in Sect. 4.

[3] ISR visualization: http://gruppe.wst.univie.ac.at/~gallm6/ISC_Viz/ISR/.
[4] ISM visualization: http://gruppe.wst.univie.ac.at/~gallm6/ISC_Viz/ISM/.

Table 1. ISC examples with specification details

Rule	Context	Requ	ISC	Specification details
①	Multi	Multi	A user is not allowed to execute more than 100 tasks (of any workflow) in a day	For the cleanup ①Ⓒ we specify a rule without a task. This rule triggers every day even when no task is executed
②	Multi	Single	Maximal KWP-2000 Connections The number of connections to KWP2000 should not exceed 10	In this case the KWP (communication protocol) allows for a maximum of 10 concurrent connections. The example shows that a set of rules can consist of different trigger positions before and after a task
③	Single	Multi	There should not exist more than one instance of W such that the input parameters (say loan customer) is the same and the loan amount sums up to \$100 K during a period of one month	Each customer is allowed to have multiple loans but in total their loan amount shall not exceed \$100 K per month. This rule set shows a timer and that multiple data elements can be used within a trigger
④	Single	Single	Wait until centrifuge is filled	We set a limit that the centrifuge is filled when 4 instances arrive. In this case we perform the action *wait* for all instances before the task *centrifugation*. After finishing the task we use a trigger without condition to perform a cleanup i.e. change variable values

- Requirement 1 is shown in rule ② and specifies a data element as *kwp _connections*.
- Requirement 2 is visualized in rule ① and sets a link to a repository where the role *user* is selected.
- Requirement 3 is the linkage of a task which is represented in all rules by either selecting a repository ① ② or a link to a specific task of one process ③ ④. We want to point out that this represents the classification of context multi ① ② and single ③ ④.
- Requirement 4, execution data, is represented in rule ④ as *counter* that counts all instances.
- Requirement 5, specification of a trigger, is shown in every rule. Some rules do not have a condition e.g. ② Ⓒ, ④ Ⓐ.

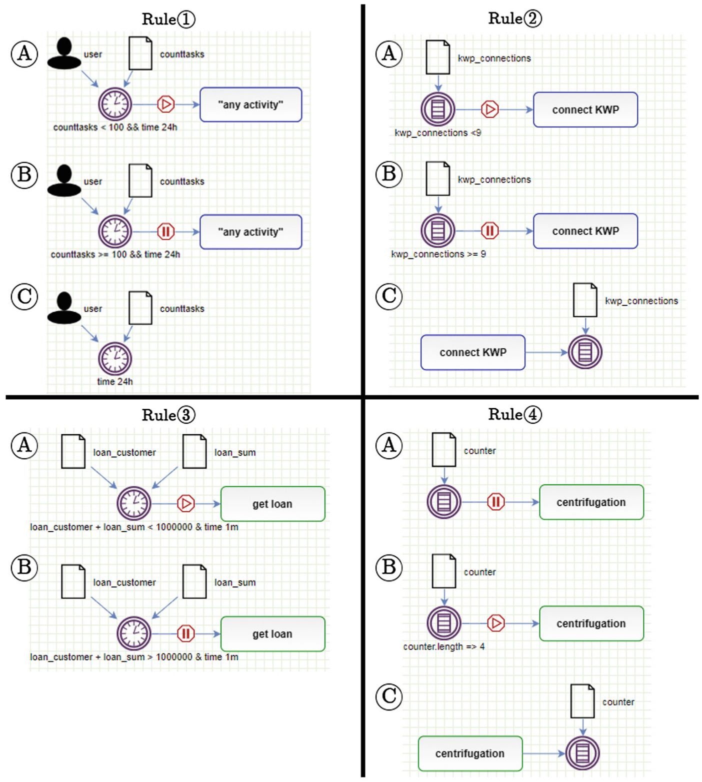


Fig. 3. Rules modeled with our extension to ISC_Viz.

– Requirement 6, actions, are visualized in nearly all examples.
– Requirement 7 and Requirement 8 cannot be seen in the visual representation, but rule numbers ①, ②, ③ and ④ could represent unique ISM IDs and the alpha numeric values Ⓐ, Ⓑ and Ⓒ represent rule IDs

Listing 1.4 shows an excerpt of the XML file exported from our tool ISC_Viz. The exported XML file is valid against the schema introduced in Sect. 5.2. The Listing specifies the centrifuge process from Table 1 ④. Every time three dots ... are shown within the listing there is some information skipped due to space limitations, this information is available on our website[5]. ISM ID (1) is followed by an ID (1) of the first rule ④Ⓐ. The rule is further specified with a priority of 11, no condition, a behaviour where the counter gets increased, and a single spanning task linked with an url and an id. Rule ④Ⓑ starts with an other ID (2), a different priority value of 7, a condition that checks if the counter is =>4, and behaviour that starts the first 4 instances if the condition is true. In the third rule ④Ⓑ the trigger is specified after the task without a condition and

[5] http://gruppe.wst.univie.ac.at/projects/crisp/index.php?t=visualization.

a behaviour that removes the instance from the counter. Our evaluation shows that we can visually model examples from each category of the classification and express them as XML.

Listing 1.4. Relax NG combination of all parts

```
...<id>1</id>             <!-- ISM ID -->
<rule><id>1</id> ...      <!-- Rule ID -->
<priority>11</priority> ...
  <condition></condition>
  <behaviour>counter &lt;&lt; event</behaviour> ...
<task>
  <spanning>single</spanning>
  <url>.../CentProcess.xml</url>
  <activityid>centrifugation</activityid>
</task>...
<id>2</id> ...            <!-- Rule ID -->
<priority>7</priority> ...
  <condition>counter.length =&gt; 4</condition>
  <behaviour>counter[0..3].each do |event    event.continue end</
      behaviour> ...
<id>3</id>                <!-- Rule ID -->
<priority>3</priority> ...
  <task> ... </task> ...
  <condition></condition>
  <behaviour>counter.shift</behaviour> ...
```

7.2 Testing the Algorithm

Algorithm 1 is applied to the example from Sect. 7.1④. The algorithm requires a priority value as input. The values in the XML file are *11*, *7*, and *3*. In the following, we decided to use the values *10*, *5* and *2* as input for the merge algorithm. The values are chosen to reflect all possible combinations. A value above 11 would result in an empty model as no rule in Listing 1.4 has a priority value of 11 or above. A value of *10* merges all rules with a value above 10, in this case the first rule. A value of *5* merges rule 1 and 2, a value of *2* merges all rules.

Figure 4 shows the visualization of the ISM from example ④. Ⓐ shows the ISM taken from our previous work [5]. Based on the selected priority Algorithm 1 creates different ISMs. Figure 4Ⓑ uses priority 10 and includes only Rule④ Ⓐ the visual differences between Rule④ Ⓐ and ISMⒷ are that the counter is now represented as *execution data*. When the priority is reduced to 5 an additional rule is included in the ISM. ISMⒸ, is identical when compared to the given ISMⒶ. If the counter is reduced to value below 3 all three rules will be incorporated into one ISM.Ⓓ depicts an ISM where two triggers are visualized. Due to space limitations we showcase all four XML files and merged models on our website[6].

ISMs created manually and ISMs created from a set of corresponding rules can differ in the number of elements depending on the priority set for Algorithm 1. Currently, Algorithm 1 creates ISM models that are complete in the sense that all visual elements will be used, meaning that all data elements, resources, trigger, and actions allowed within an ISM will be visualized.

[6] http://gruppe.wst.univie.ac.at/projects/crisp/index.php?t=visualization.

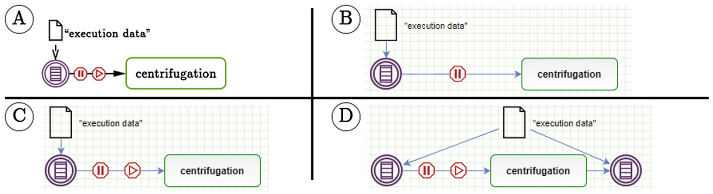


Fig. 4. ISM visualization from the files created by Algorithm 1.

8 Conclusion

Compliance management demands to bridge the gap between models of instance-spanning constraints (ISM) and their executable rules (ISRs). Visual representations of constraints are used for reflecting and discussing regulations, but can be specified in different ways for implementing constraints across multiple instances. This work provides a visual representation including a modeling tool for the specification of ISRs as well as the export and further specification of XML representations. Moreover, merging ISRs into an ISM is enabled. In future work, we will examine the automatic creation of ISMs based on ISRs, i.e., for creating ISMs without the need of unique IDs. Furthermore we want to evaluate how to visually model the behaviour part of a trigger.

Acknowledgment. This work has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT15-072.

References

1. Awad, A., Weidlich, M., Weske, M.: Visually specifying compliance rules and explaining their violations for business processes. J. Vis. Lang. Comput. **22**(1), 30–55 (2011)
2. Awad, A.: BPMN-Q: a language to query business processes. In: Proceedings of EMISA 2007, pp. 115–128 (2007)
3. Becker, J., Ahrendt, C., Coners, A., Weiß, B., Winkelmann, A.: Business rule based extension of a semantic process modeling language for managing business process compliance in the financial sector. In: GI Jahrestagung (1), pp. 201–206 (2010)
4. Fdhila, W., Gall, M., Rinderle-Ma, S., Mangler, J., Indiono, C.: Classification and formalization of instance-spanning constraints in process-driven applications. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 348–364. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_20
5. Gall, M., Rinderle-Ma, S.: Visual modeling of instance-spanning constraints in process-aware information systems. In: Advanced Information Systems Engineering, pp. 597–595, May 2017
6. Ghose, A., Koliadis, G.: Auditing business process compliance. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 169–180. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74974-5_14
7. Heinlein, C.: Workflow and process synchronization with interaction expressions and graphs. In: International Conference on Data Engineering, pp. 243–252 (2001)

8. Indiono, C., Mangler, J., Fdhila, W., Rinderle-Ma, S.: Rule-based runtime monitoring of instance-spanning constraints in process-aware information systems. In: Debruyne, C., Panetto, H., Meersman, R., Dillon, T., Kühn, E., O'Sullivan, D., Ardagna, C.A. (eds.) OTM 2016. LNCS, vol. 10033, pp. 381–399. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_22
9. Leitner, M., Mangler, J., Rinderle-Ma, S.: Definition and enactment of instance-spanning process constraints. In: Wang, X.S., Cruz, I., Delis, A., Huang, G. (eds.) WISE 2012. LNCS, vol. 7651, pp. 652–658. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35063-4_49
10. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.P.: Compliance monitoring in business processes: functionalities, application, and tool-support. Inf. Syst. **54**, 209–234 (2015)
11. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Design and verification of instantiable compliance rule graphs in process-aware information systems. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 9–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13094-6_3
12. Mangler, J., Rinderle-Ma, S.: IUPC: identification and unification of process constraints. CoRR abs/1104.3609 (2011). http://arxiv.org/abs/1104.3609
13. Moody, D.: The physics of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Softw. Eng. **35**(6), 756–779 (2009)
14. Pufahl, L., Herzberg, N., Meyer, A., Weske, M.: Flexible batch configuration in business processes based on events. In: Franch, X., Ghose, A.K., Lewis, G.A., Bhiri, S. (eds.) ICSOC 2014. LNCS, vol. 8831, pp. 63–78. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45391-9_5
15. Rinderle-Ma, S., Gall, M., Fdhila, W., Mangler, J., Indiono, C.: Collecting examples for instance-spanning constraints. Technical report, CoRR abs/1603.01523 (2016)
16. Warner, J., Atluri, V.: Inter-instance authorization constraints for secure workflow management. In: Symposium on Access Control Models and Technologies, pp. 190–199 (2006)
17. Wieringa, R.J.: Design Science Methodology for Information Systems and Software Engineering. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43839-8
18. Winter, K., Rinderle-Ma, S.: Discovering instance-spanning constraints from process execution logs based on classification techniques. In: 21st IEEE International Enterprise Distributed Object Computing Conference, pp. 79–88 (2017)

Improving the Usability of Process Change Trees Based on Change Similarity Measures

Georg Kaes(✉) and Stefanie Rinderle-Ma

Faculty of Computer Science, University of Vienna, Vienna, Austria
{georg.kaes,stefanie.rinderle-ma}@univie.ac.at

Abstract. Flexible process management systems store information about conducted process change operations in change logs. Change log analysis can provide users who are responsible for planning and executing upcoming adaptations with valuable information. Change trees represent change logs emphasizing the temporal relation between change operations such that users can immediately see which change sequences have been applied in the past. Similar to most process mining approaches, change trees currently build upon label equivalence. However, labels only provide restricted information about a change operation. Hence this paper investigates how process change similarity can be employed to compare changes, i.e., similar change operations are aggregated in the tree as they appear in a change sequence. A user experiment shows the increased efficiency of the aggregated change sequences: users find relevant information faster than in a change tree based on label equivalence.

1 Introduction

Flexible business process management systems allow users to change their processes according to the current requirements [1]. Information about the conducted change operations are stored in a change log, which can be used to analyze past change operations.

For analyzing change logs, different approaches have been developed: While *change processes* [2] focus on causal dependencies between process change operations, the *change tree* [3] focuses on temporal relations between change operations, i.e. the inherent change sequences. Using a change tree, one can see in which ways a certain kind of process instance has evolved, and which evolution paths are more frequently used than others. Analyzing change sequences can provide valuable information: Imagine a nursing home with hundreds of patients, where each patient's therapy plan is represented by a process instance. Each task in this process instance represents an activity which has to be completed in order to improve the health status of a patient. Whenever an adaptation to a therapy instance is required (e.g., because the patient feels sick), his process instance has to be adapted, thus generating a new change in his change sequence. Information about the change sequence of a specific patient can be used to analyze what has been done, e.g., when evaluating the effectiveness of a patient's therapy plan. Following, it can be analyzed whether the patient's treatment was unique, or similar to other, comparable situations.

Change trees are a data structure based on process change logs which emphasize such change sequences. They are constructed by aggregating equal change operations on the same position of change sequences in a single node, and generating sibling nodes for different change operations. However, change trees compare process change operations based on their label only. If two change operations have the same label, they are considered to be equal. For many situations, such an assumption proves to be too general. Depending on the analysis question at hand, for example, it could be more interesting to compare process change operations based on the required resources, temporal constraints, or other attributes of the change operation. In other words, comparing change operations shall be shifted from *label equivalence* towards comparing their *semantics*.

The idea behind is to replace label equivalence with *change similarity measures* (cf. [4]). They calculate the similarity of two process change operations from different change perspectives [4], e.g., change attributes or change effects on the adapted process instance. This enables to compare change operations regarding, e.g., the required resources, time constraints, or affected control and data flow.

For example, the *Change Resource Similarity (CRS)* [4] calculates the similarity of the required resources of two change operations which insert activities into a process instance in an interval between 0 and 1: If two inserted process fragments require e.g. 3 nurses each, CRS returns 1, if they require totally different resources, it returns 0. If a nurse wants to know which resources are usually required for the treatment of a specific disease, she could investigate the change tree where similar change operations regarding CRS are aggregated in one node.

Replacing label equivalence by change similarity measures poses several challenges. If changes are similar, different options for merging them in the resulting change tree become possible. This necessitates theoretical considerations on how a change tree can be built in this case. Moreover, the interpretation of a change tree based on similarity measures is different from one based on label equivalence. Finally, it has to be investigated which benefits arise from employing change similarity instead of label equivalence for change trees. This leads to the following research questions:

RQ1: How can change sequences be analyzed in a semantic way? Which measures can be used? How do these measures have to be applied to a set of change sequences?

RQ2: Does the aggregation of change operations in change trees help the user to find relevant information faster than in change trees based on label equivalence?

In this paper, we first analyze how similarity measures for process change operations can be applied to sets of change sequences as they can be found in a change log, thus allowing us to answer different questions about change sequences

($\rightarrow$ RQ1). This is followed by an evaluation with potential users of the change tree ($\rightarrow$ RQ2).

This paper is structured as follows: Sect. 2 introduces the basic notations which are required for the remainder of the paper. This is followed by a general introduction to the application of process change operation similarity to change trees (Sect. 3). In Sect. 4, we show how similar and equal change operations can be aggregated. In our evaluation (Sect. 5) we present an experiment conducted with over 60 users who analyzed change sequences using our approach. The paper concludes with related work (Sect. 6) and a summary (Sect. 7).

2 Fundamentals

This paper focuses on the control flow aspects of process changes and defines process schema S as S := (N,E), where N denotes the set of nodes (activities and gateways), and E denotes the set of control flow edges (c.f. [5]). A change operation Δ transforms process schema S into adapted schema S', formally:

Definition 1 (Process Change Operation). *A process change operation Δ is defined as a tuple $\Delta := (t, f, p, S)$ where*

- *t denotes the type of the change operation.*
- *f denotes the process fragment which is used by the change operation.*
- *p denotes the position of the change operation.*
- *S denotes the process schema or instance the change operation is applied to.*

During runtime changes are applied to a set of process instances I running on a schema S. These changes are stored in a *change log* $\mathcal{C}$. Specifically, for each $i \in I$, $\mathcal{C}$ stores the temporally ordered set of change operations that have been applied to i. An example of a change log is given in the left pane of Fig. 1.

A *change tree* is a data structure representing change log information and is defined as follows:

Definition 2 (Change Tree, Based on [3]). *Let $\mathcal{C}$ be a change log and $\mathcal{D}$ be the set of all change operations contained in $\mathcal{C}$. Then the change tree T is defined as a rooted multiway tree $T := (r, V, E)$ with*

1. *$r := \emptyset$ is the unique root node*
2. *$V \subseteq \mathcal{D} \times \mathbb{N}_0$*
3. *$E \subseteq V \times V$*
4. *$\forall$ paths p from root r to node $v = (\Delta, n) \in V$ with $n > 0$: p corresponds to the change traces of n changed process instances in $\mathcal{C}$.*
5. *$\forall$ leaf nodes $v = (\Delta,n) \in V$: $n > 0$*

The change tree is constructed from a process change log. The root node represents the unchanged process schema. Each change operation which has been applied on a process instance is represented by a node in the tree. The traces in the change tree from the root nodes to the leaf nodes represent the temporal relationships between the applied change operations: If Δ_1 has been applied before Δ_2, it is closer to the root node.

Figure 1 shows an example for a change tree based on a process change log that consists of six traces $t_1, t_2, \ldots, t_6$. There are only two ways in which the process instances have been adapted in the beginning, i.e., Δ_1 and Δ_3. Following Δ_1, there is one trace where Δ_2 has been applied, and two traces where Δ_1 has been applied again. Both traces end at this point. Each node in a change tree contains two values: The first value is the label of the change operation which has been applied at this point in the change traces, the second value represents the number of change sequences which terminate at this change operation. This can be checked by consulting the change log, where indeed two change traces contain two applications of Δ_1, and one trace contains Δ_1 followed only by Δ_2.

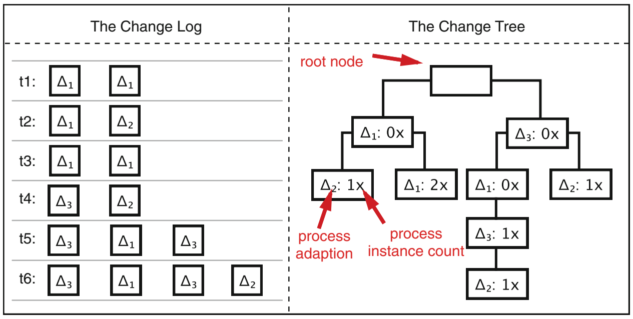


Fig. 1. Change tree representation of an example change log

The structure of the change tree emphasizes multiple features of the change log it is based on: The *breadth of the change tree* is an indicator for the diversity of the applied process change operations. The broader the change tree, the more diverse the change operations have been. The *height of the change tree* indicates how many change operations have been applied to one or more change traces. The number of change traces in a certain branch in relation to the total number of change traces in the log shows which change operation paths have been used more frequently. The diversity of the change operations which have been applied on a certain level can be seen from the number of *sibling nodes*.

So far, in a change tree, two (potentially sibling) change operations have only been aggregated in one node they were equivalent regarding their labels; the attributes and effects of the change operations were not considered at all. Using label equivalence has been discussed as a potential limitation in the area of process mining (cf. e.g., [6]). For example, it would be possible that two activities have equal labels, but launch different actions based on the context they are executed in, or that two activities with different labels are actually semantically equivalent (i.e., the launched actions are the same). Hence, for certain analysis questions, using an equivalence notion that refers to the semantics of activities would be useful [7]. The same holds for comparing change operations.

Hence, this work proposes the aggregation of nodes in change trees based on process change similarity instead of using label equivalence. In the following, we denote the similarity of two change operations $\varDelta_1$ and $\varDelta_2$ as $sim(\varDelta_1, \varDelta_2) \in [l, 1]$. For some similarity measures, $l = 0$ holds, for others $l = -1$ is defined. Which similarity measure is of interest, depends on the question at hand. When resources required to complete activities are sparse (e.g., nurses in the nursing domain, or professionals in the production domain), a resource-based view on the change log might be of interest. Using such a resource-based view, one can see more easily which resources have been required, and based on this information, it might be possible to optimize future process adaptations. For such a situation, the *Change Resource Similarity (CRS)* [4] can be used: It compares the resources used by the fragments of two change operations $\varDelta_1$ and $\varDelta_2$, which are inserted into or deleted from a process schema. If the resources are exactly the same, and both change operations insert or delete, it returns 1. If the resources are exactly the same, but one change operation inserts, and the other deletes, it returns -1. If the required resources are different, it returns a value between $(-)1$ and 0. Contrary, the attribute based similarity measure $sim_{attr}(\varDelta_1, \varDelta_2)$ returns a value in the range of $[0, 1]$ [4]. It is calculated by defining the similarity for each attribute of the two process change operations (fragments, positions, operation types and schemas), and weighing the results. Using weights enables to prefer certain attributes when comparing the change operations. Attributes can be even ignored by assigning a weight of 0 to them.

Definition 3 (Process Change Operation Similarity Measure). *Let $\varDelta_1$ and $\varDelta_2$ be two change operations as defined in Definition 1. Further, let $l \in \mathbb{R}$, $l < 1$. Then, the similarity measure is defined as $sim(\varDelta_1, \varDelta_2) \in [l, 1]$.*

Which process change operation similarity measure is used highly depends on the analysis question: If resources are of interest, a resource-based similarity metric such as CRS may be useful; if control-flow related features are of interest, a more attribute-based measure might be favorable. However, the approach presented in this paper abstracts from the implementation of concrete similarity measures. Thus, any measure which calculates a similarity $sim(\varDelta_1, \varDelta_2) \in [l, 1]$ can be used as a basis for creating the aggregated change tree.

3 Applying Similarity Measures to Change Logs

To discriminate between traditional change trees (based on label equivalence) and the change trees based on similarity measures as presented in this paper, we will refer to the latter by the term *aggregated change tree*. The term stems from the fact that multiple similar change operations will be *aggregated* in a single node. Figure 2 depicts an example of a change log, a change tree and an aggregated change tree. The middle pane shows a change tree according to Definition 2 based on the change log in the left pane: It consists of two levels, where the first level contains the four change operations $\varDelta_1$, $\varDelta_2$, $\varDelta_3$ and $\varDelta_4$. The change traces t_2 and t_3 additionally contain a second change operation, $\varDelta_1$. The right pane depicts the *aggregated change tree*. In the following sections, we will discuss how this aggregated change tree is created.

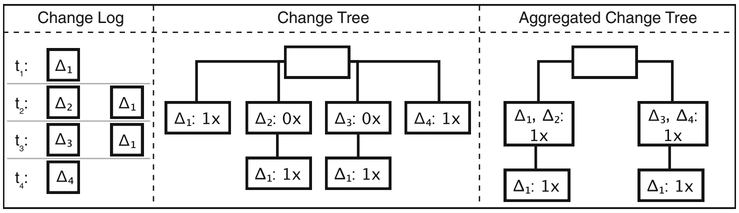


Fig. 2. Basic example

To preserve the change sequences of a change log, which are represented in a change tree, only change operations are considered for aggregation which would be on the same level of the change tree. While it would be possible to aggregate change operations in an aggregated change tree vertically (by aggregating change operations which happened consecutively), the horizontal aggregation as presented in this paper preserves the basic properties of the change tree: As the change tree, the aggregated change tree should, when read from the root to the leafs, represent the set of change sequences from a process log. If consecutive change operations would be aggregated as well, this property would be lost.

The only difference between the aggregated change tree lies within the structure of the nodes. Instead of a node V containing a single change operation (as defined in Definition 2), V now contains a set of similar change operations:

Definition 4 (Aggregated Change Tree). *Let $\mathcal{D}$ be a set of change operations as defined in Definition 2. An aggregated change tree $ACT := (r, V, E)$ is a change tree $T := (r, V, E)$ where $V \subseteq 2^{\mathcal{D}} \times \mathbb{N}_0$.*

There are two border cases for the number of nodes present in the aggregated change tree (ACT) compared to the change tree: Due to the horizontal aggregation, the minimum number of nodes equals the depth of the original change tree, since the ACT contains only one node for each level of the change tree. If however no nodes can be aggregated, the ACT contains exactly as many nodes as the change tree.

In order to decide whether two change operations Δ_1 and Δ_2 should be aggregated in a single node, a similarity threshold value $\tau \in [l, 1]$ is specified where l corresponds to the lower bound of the similarity measure of interest. If the similarity measure $sim(\Delta_1, \Delta_2) \geq \tau$, Δ_1 and Δ_2 are aggregated in a single node; else, two separate nodes are created. In the next section, we will present two approaches for generating aggregated change trees based on a similarity measure, a threshold τ, and a change log.

4 Generating the Aggregated Change Tree

Depending on the chosen similarity threshold τ, the aggregated change tree can be created in different ways. Section 4.1 presents the algorithms for $\tau < 1.0$ and Sect. 4.2 follows up with an illustration for $\tau = 1.0$.

4.1 Aggregation for Similar Process Change Operations

When aggregating similar change operations in one node, the similarity threshold τ has to be reached for all similarity measures between the change operations in the aggregation node. Figure 3 shows an example of such an aggregation: In pane 1, a change log is depicted which contains three change traces consisting of the process change operations Δ_1, Δ_2 and Δ_3. Pane 2 shows the similarity of the three change operations: $sim(\Delta_1, \Delta_2) = 0.75$, $sim(\Delta_2, \Delta_3) = 0.8$ and $sim(\Delta_1, \Delta_3) = 0.6$. Pane 5 shows the change tree which would be generated based on label equivalence (c.f. [3]). Depending on the chosen similarity threshold τ, different change operations can be aggregated in one node: If $\tau = 0.6$, all three change operations can be aggregated in one node (c.f. aggregated change tree depicted in pane 6). If $\tau = 0.7$, only change operations Δ_1 and Δ_2, or change operations Δ_2 and Δ_3 can be aggregated in one node, but not Δ_1, Δ_2 and Δ_3, since the similarity between Δ_1 and Δ_3 is below τ. The similarity measure is not transitive. Thus, for $\tau = 0.7$, two aggregated change trees are possible (c.f. panes 7 and 8). These two change trees are the results of two different aggregation strategies, which are explained at the end of this section.

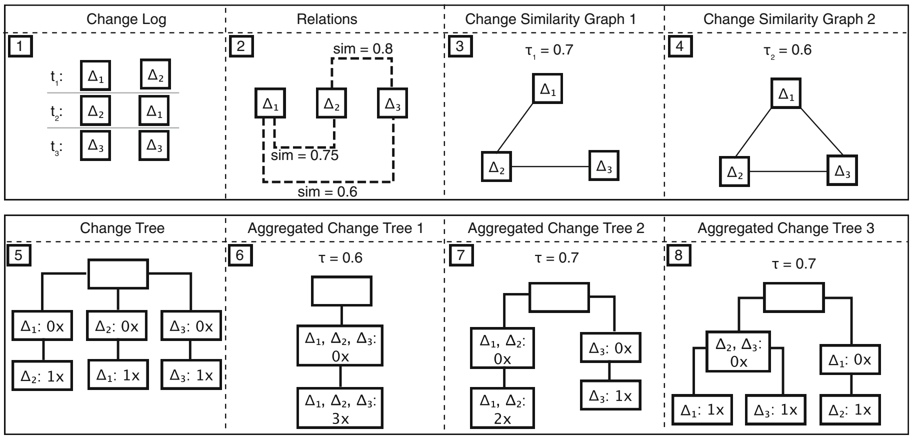


Fig. 3. Transitivity example

The similarity measures between all change operations which can be aggregated in one node need to be above the threshold τ. In order to find the subset of change operations for which this condition holds, we introduce the *change similarity graph*. (c.f. Definition 5).

Definition 5 (Change Similarity Graph). *Let $\mathcal{D}$ be a set of change operations, $sim(\Delta_i, \Delta_j)$ be a similarity measure, and τ the similarity threshold. The Change Similarity Graph $\Gamma := (\mathcal{D}, E)$ is an undirected graph where $E \subseteq \mathcal{D} \times \mathcal{D}$ denotes the set of edges. There exists an edge between change operations Δ_i, $\Delta_j \in \mathcal{D}$, iff $sim(\Delta_i, \Delta_j) \geq \tau$.*

Pane 3 and 4 of Fig. 3 show the similarity graphs for two thresholds $\tau_1 = 0.7$ and $\tau_2 = 0.6$ for the process change operations Δ_1, Δ_2 and Δ_3 from the example given above. In the similarity graph for $\tau_1 = 0.7$, there exist only edges between change operations Δ_1 and Δ_2, and between Δ_2 and Δ_3. There is no edge between Δ_1 and Δ_3, since its value lies below the threshold at 0.6.

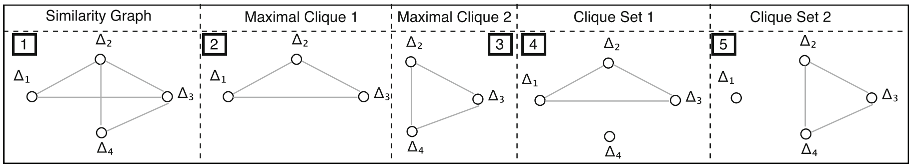


Fig. 4. Set of maximal cliques in a change similarity graph

Using the similarity graph as a basis, the problem of finding the subset of change operations where the similarity of all change operations exceeds the threshold τ can be seen as the problem of finding all maximal cliques in the graph. A maximal clique in an undirected graph is a *maximal complete subgraph* [8], that is a set of nodes and edges from that graph, where there exists an edge between each node. Using a set of cliques as a basis, one could aggregate all change operations which are in the same clique, since the similarity of those change operations exceeds τ.

The graph depicted in the third pane of Fig. 3 ($\tau = 0.7$), contains two cliques: One contains change operations Δ_1 and Δ_2, the other contains change operations Δ_2 and Δ_3. In the graph in the fourth pane ($\tau = 0.6$), there exists one clique which contains all three change operations. The *Bron-Kerbosch algorithm* [8] takes an undirected graph such as the change similarity graph as input, and returns all maximal cliques of this graph. Figure 4 shows an example for a set of maximal cliques in a change similarity graph. Pane 1 shows the similarity graph which contains four different process change operations. By applying the Bron-Kerbosch algorithm [8], two maximal cliques (depicted in panes 2 and 3) are generated. Obviously, Δ_2 and Δ_3 appear in more than one clique. If both cliques were used directly to generate the nodes of the aggregated change tree, Δ_2 and Δ_3 would appear in more than one node.

Following, the similarity graph has to be split up into a *unique clique set*, containing several cliques where each node is present in exactly one clique. This can be done in a straightforward fashion: We start with a similarity graph $\mathcal{G}$ and an empty unique clique set $\mathcal{U}$. In the first step, the cliques of $\mathcal{G}$ are created, all nodes contained in the biggest maximal clique in $\mathcal{G}$ are removed from the graph, and this biggest maximal clique is added to $\mathcal{U}$. This step is now repeated with the remaining nodes of the graph, until no nodes are left in the graph. If there exist multiple biggest maximal clique sets, multiple solutions are possible, as seen in the following example:

Applying this procedure to the example given in Fig. 4 yields the following results: We start with an empty unique clique set $\mathcal{U} = \emptyset$. If we start with the first maximal clique $\{\Delta_1, \Delta_2, \Delta_3\}$ (depicted in pane 2) this clique is added to $\mathcal{U}$, and these nodes are removed from the similarity graph. Following, the similarity graph now only contains $\{\Delta_4\}$, which is the last maximal clique left in the graph; thus, it is added to $\mathcal{U}$, resulting in the unique clique set depicted in pane 4. If we start with the second maximal clique set $\{\Delta_2, \Delta_3, \Delta_4\}$ (depicted in pane 3), $\{\Delta_1\}$ would be the second clique, and we would end up with the unique clique set depicted in pane 5. Thus, two unique clique sets are created.

These two unique clique sets can now be used to generate the change tree nodes. Depending on the which set is chosen, either Δ_1, Δ_2 and Δ_3 are aggregated in one node, and Δ_4 is in the other node, or Δ_2, Δ_3 and Δ_4 are aggregated in a node, and Δ_1 is in its own node. Two distinct features can be used as a basis for deciding which process change operation should be aggregated in a node: (a) the most similar change operations should be aggregated or (b) the number of nodes in the tree should be minimized.

Figure 5 shows a change log with four change operations Δ_1 to Δ_4. Pane 2 depicts the similarity relations between them. In the remainder of this section, this example will be used to explain the different aggregation strategies.

Aggregating the Most Similar Change Operations. Algorithm 1 shows the strategy which can be used to aggregate the most similar change operations on one level into an aggregated change tree node. It is based on the creation for change trees as defined in [3] and starts with the first level of change operations in the log, containing Δ_1 to Δ_4 (c.f. example from Fig. 5).

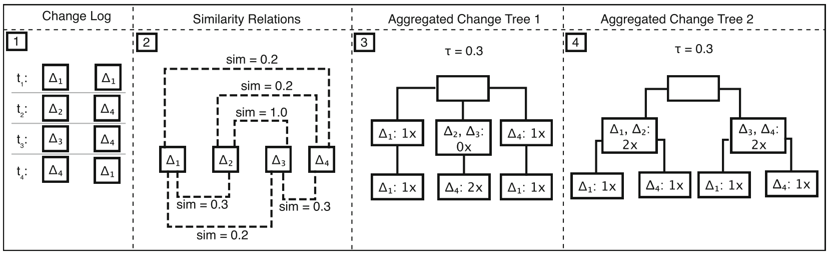


Fig. 5. Example change tree for different strategies

First, the unique clique sets are created. For a $\tau = 0.3$, two unique clique sets are created: $\{\{\Delta_1\}, \{\Delta_2, \Delta_3\}, \{\Delta_4\}\}$ and $\{\{\Delta_1, \Delta_2\}, \{\Delta_3, \Delta_4\}\}$. The optimal clique set is defined as the unique clique set where the overall similarity of the contained nodes is maximal. This is calculated and compared to the similarity of other unique clique sets in the function *get-optimal-cliqueset*. Initially, it iterates over a set of unique clique sets. For each unique clique set, if a clique contains more than one Δ, the similarity of all change operations in this clique is calculated, and its sum is divided by the number of total nodes in the clique. For the first clique set containing 3 cliques, the similarity of Δ_2 and Δ_3 is 1.0, which results in an overall similarity of $\frac{1}{3}$. For the second unique clique set containing 2 cliques, the similarity is 0.3 for both cliques, thus, the resulting similarity is also 0.3. Since $\frac{1}{3} > 0.3$, the first clique set $\{\{\Delta_1\}, \{\Delta_2, \Delta_3\}, \{\Delta_4\}\}$ is chosen. Based on this clique set, for each clique, a node is created and the traces are added for creating the child nodes with the change operations in the next recursive iterations. For each change operation in the clique, the corresponding change trace is checked: If the change operation is the last change operation of this trace, the node's counter is incremented; else, the trace is added to the traces, which will be considered in the next recursive call. Thus, all relevant traces and change operations will be considered for the child level. The resulting aggregated change tree is presented in pane 3 of Fig. 5.

Algorithm 1. Aggregating the most similar change operations

```
Input:
  – Change log $\mathcal{C}$
  – Similarity measure $sim(\Delta_1, \Delta_2)$
  – change similarity graph $\Gamma := (\mathcal{D}, E)$

Output: Aggregated Change Tree ACT
Begin root = create-node(null,null);
create-children(root, $\mathcal{C}$, 0); return $\mathcal{ACT}$
function create-children(parent,traces,position)
    nodes = $\emptyset$, deltas = $\emptyset$, cliquesets = $\emptyset$
    foreach trace $\in$ traces do
        deltas = deltas $\cup$ {trace.get-changeoperation(position)}
    generate-cliqueset($\mathcal{D}$, $\emptyset$, cliquesets)
    //variable cliquesets now holds the set of clique sets
    optimal-cliqueset = get-optimal-cliqueset(cliquesets)
    //variable optimal-clique set now holds the optimal clique set
    foreach clique $\in$ optimal-cliqueset do
        //creates a new node containing the change operations and links it to the parent node
        node = create-node(clique.changeoperations, parent)
        nodes = nodes $\cup$ {node}
        foreach delta $\in$ clique do
            //get-trace-of(delta) finds the trace this change operation is contained in
            trace = get-trace-of(delta)
            if trace.get-changeoperation(position+1)==null then
                node.count++
            else
                node.nexttraces = node.nextraces $\cup${trace}
    foreach node in nodes do
        create-children(node,node.nexttraces,position+1)
function get-optimal-cliqueset(cliquesets)
    highest-sim = 0, optimal-set = $\emptyset$
    foreach cliqueset $\in$ cliquesets do
        sim = 0
        foreach clique $\in$ cliqueset do
            if |clique| > 1 then
                //calculate similarity in clique
                sim = $\frac{\sum_{i,j \in clique, i \neq j} sim(i,j)}{|clique|}$
        if optimal-set == $\emptyset \vee$ sim > highest-sim then
            highest-sim = sim, optimal-set = cliqueset
```

Minimizing the Number of Nodes. The second strategy minimizes the number of nodes in each level of the aggregated change tree. Algorithm 2 shows this strategy. It replaces the function *get-optimal-cliqueset* with a different algorithm which prefers the unique clique set with the minimum number of cliques. Since each clique is represented by a node in the aggregated change tree, this results in the lowest number of nodes *per level*. For the first call of the function in the example given in Fig. 5, this would return the clique set $\{\{\mathit{\Delta}_1, \mathit{\Delta}_2\}, \{\mathit{\Delta}_3, \mathit{\Delta}_4\}\}$, thus resulting in the aggregated change tree depicted in pane 4.

Algorithm 2. Generating the minimum amount of nodes per level

```
function get-optimal-cliqueset(cliquesets)
    min-count = 0, optimal-set = $\emptyset$
    foreach cliqueset $\in$ cliquesets do
        if optimal-set == $\emptyset \vee$ min-count < |cliqueset| then
            min-count = |cliqueset|, optimal-set = cliqueset
```

Discussion: We have presented two strategies for aggregating similar change operations in change tree nodes. Which one should be used highly depends on the structure of the change log: For most situations, aggregating the most similar change operations in one node seems logical. In the nursing home, for example, one could aggregate the most similar therapy adaptations in a single change tree node. If however the resulting number of nodes would become very large due to the structure of the change log, minimizing the number of change tree nodes instead can enhance the tree's readability, thus simplifying further analysis of the change log.

4.2 Aggregation for Equal Process Change Operations

For the aggregation of equal change operations ($\tau = 1.0$), no change similarity graph is required, since the equality relation is transitive ($\mathrm{sim}(\mathit{\Delta}_1, \mathit{\Delta}_2) = 1.0 \wedge \mathrm{sim}(\mathit{\Delta}_2, \mathit{\Delta}_3) = 1.0 \implies \mathrm{sim}(\mathit{\Delta}_1, \mathit{\Delta}_3) = 1.0$). The example we use to explain the aggregation of equal change operations is based on the change log as presented in Fig. 2[1]. While the middle pane shows the change tree based on label equivalence as defined in [3], the right pane shows the resulting aggregated change tree. Assume that $sim(\mathit{\Delta}_1, \mathit{\Delta}_2) = 1.0$, $sim(\mathit{\Delta}_3, \mathit{\Delta}_4) = 1.0$, and all other similarities <1.0. The aggregated change tree is created as follows: First, an empty root node with no label and no parent is created. Next, each "column" in the change log is iterated over: The first column contains change operations $\mathit{\Delta}_1, \mathit{\Delta}_2, \mathit{\Delta}_3$ and $\mathit{\Delta}_4$. The second column contains $\mathit{\Delta}_1$ (following $\mathit{\Delta}_2$ respectively $\mathit{\Delta}_3$). For the first column, the similarities between all change operations are checked: If $sim(\mathit{\Delta}_i, \mathit{\Delta}_j) = 1.0$, the change operations are aggregated in one node - if not, they are split up in two nodes. In this example $sim(\mathit{\Delta}_1, \mathit{\Delta}_2) = 1.0$ and

[1] The full version of the algorithm can be found on our project homepage https://cs.univie.ac.at/project/APES.

$sim(\Delta_3, \Delta_4) = 1.0$. All other similarities are smaller than 1. The same is done for the second column. Note that the Δ_1 of the second column cannot be aggregated in one node, since they have distinct parent nodes (one containing Δ_1, Δ_2, and one containing Δ_3, Δ_4). Ultimately the aggregated change tree depicted in pane 3 of Fig. 2 is created.

5 Evaluation

The approach presented in this paper is evaluated in an experiment which aims at two goals, i.e., to analyze whether (a) the change tree can be used as a basis for analyzing change sequences and (b) certain questions can be answered faster by participants using an aggregated change tree, than using a change tree.

Method and Experiment Design. The experiment is based upon a prototypical implementation of the (aggregated) change tree, working in every modern browser using HTML, CSS and JavaScript. This implementation serves as a proof of technical feasability, which can also be accessed without participating in the experiment[2].

The first draft of the experiment was refined using a two stage pretest: During stage one, the questions of the experiment were discussed with three peers who are familiar with the concept of process change operations and change trees. In the second phase, persons from the target audience were asked to participate in the experiment. The target audience of the aggregated change tree are persons from different fields of work, who do not necessarily have to be familiar with process change operations. Thus, the chosen participants were older than 18 years, and were not required to have any knowledge regarding process change operations, or change trees. During the pretest, two major parts of the experiment were optimized: First, the user interface was optimized to be more intuitive to users who are not familiar with the setting, second, the wording of two questions and their explanation was optimized.

The experiment consists of two parts: Part 1 addresses the question whether the change tree (CT) is a better representation of change logs than their raw XML format (XML). Specifically, we want to find out if certain questions regarding the change log (e.g. *How many resources have been used in this change log?*) can be answered faster using a change tree than an XML based log file. The participants were split in two groups and had to answer the same questions based on those two representations. In each question, we measured the time until the (correct) answer was found. The questions for the two representations were based on different, but similar log files to prevent bias due to the participants knowing the correct answer: While group 1 answered questions for the CT based on log A, and questions for XML based on log B, group 2 received the CT questions based on log B, and the XML questions based on log A. This setup also helped us to mitigate the risk of a bias due to different log files: If there was only one

[2] The prototype, the experiment including all questions, and all related data can be found at https://cs.univie.ac.at/project/APES.

group who answered all CT questions based on log A, and all XML questions based on log B, the differences in speed and correctness could be due to the slight differences between those log files, and not due to their representation. To mitigate the risk of distortion due to a learning effect, the order of the questions and representations was randomized. Part 2 of the experiment addresses the usability of the aggregated change tree (ACT) in comparison to the CT. The setup and the participants were the same as in Part 1; however, two log files C and D (different from those of Part 1) were used.

At the beginning of the experiment, the participants were introduced into the topics of processes, process adaptations, change logs and (aggregated) change trees. Following this 10 min presentation, the participants had the possibility to ask questions, and received a link to the experiment. They had three days to participate in the experiment. Such an *uncontrolled experiment setup* resembles a realistic setting, in which aggregated change trees would be used, more closely.

Results: In total, 64 participants participated in the experiment. These participants were divided in two groups as described above. 31 participated in group 1, and 33 participated in group 2. First, the data set was cleaned for participants who obviously entered random numbers. The check was based on two parameters: (1) most of the answers were wrong, and (2) all answers were given in less than two seconds. Answers given that quickly can be seen as an indicator for a participant entering wrong answers on purpose. These patterns were found for one participant from group 1, resulting in 30 valid participants in group 1, and 33 valid participants in group 2.

Table 1. Results of the experiment

XML Log vs CT	Overall		Group 1		Group 2	
	Log	CT	Log	CT	Log	CT
Correct answers	61.11%	68.25%	61.67%	68.33%	60.61%	68.18%
Mean time (sec.)	90.02	35.43	95.28	33.12	83.37	39.97
Median time (sec.)	58.69	29.2	57.51	27.32	59.43	32.14
CT vs ACT	Overall		Group 1		Group 2	
	CT	ACT	CT	ACT	CT	ACT
Correct percentage	65.08%	67.46%	71.67%	68.33%	59.09%	66.67%
Mean time (sec.)	112.54	43.97	138.08	52.84	84.39	35.71
Median time (sec.)	51.73	22.43	61.82	24.18	48.28	21.4

Table 1 summarizes the results from the experiment. The mean and median times refer to the amount of time needed by the participants to find the correct answers. Using a two sample t test, the differences regarding the amount of correct answers given are not significant ($p > 0.05$): the correctness of the answers was not negatively influenced when using CT compared to the XML Log and

ACT compared to CT respectively. However, the time required to find the correct answer is statistically significant for both parts of the experiment ($p < 0.05$): In the first part of the experiment, it can be seen that the mean and median times required to find the correct answer for the CT is much lower than for the XML; in the second part, it can be seen that participants using the ACT require even less time to find the correct answer than participants who use the CT.

Discussion and Threats to Validity: The experiment shows that the (A)CT can be used as a representation of change logs, since the amount of correct answers is not statistically different from the amount in the XML Log. This effect may also be due to the XML Log being very simple: For Part 1 of our experiment we chose a log with only four change operations in total, in order to not demoralize participants who had to work with the XML representations. Thus, the change log was designed in favor of the XML representation, which may pose a threat to the validity. However, even in such a setting, the time required for the correct answers was statistically significantly lower for CT than for XML Log. For the ACT, the number of correct answers is not statistically significantly different from CT, but significantly less time is required to find a correct answer. Although the uncontrolled setup of our experiment increases the resemblance of our experiment to a realistic setting in contrast to a controlled setup, this decision also poses a threat to validity, since participants have been able to communicate with each other while participating. However, participants who completed the experiment did not receive the correct answers to the questions, and such behavior may also be found in a realistic setting, where users would have to analyze change trees. Thus, the results can be seen as being more realistic, while the negative impact of the threat has been kept to a minimum.

6 Related Work

In [9], the authors present an overview over existing approaches to determine the similarity of two process models. The presented approaches range from label matching similarity, over feature-based similarity estimations up to comparisons of common nodes and edges in process model graphs. These approaches can also be interesting for change operation similarity measures, since they can be used to compare attributes of change operations, like the schema, the applied fragment, or the position of the change. *Van Dongen et al.* discuss causal footprints, *which is a collection of the essential behavioral constraints imposed by a process model* [10] as a representation of a process model's behavior. Using causal footprint's *look-back* and *look-ahead* links, one could analyze and compare the positions of process change operations.

The work presented in [11] provides similarity measures for process instances. To the best of our knowledge, [4] is the only approach dealing with process change similarity. This work exploits the measures proposed in [4], but any other change similarity measure can be used as well.

In contrast to change trees, change processes [2] are a different method to analyze process change logs. Change processes focus on causal relations between process change operations. If a change operation Δ_1 is followed by Δ_2, it means that Δ_2 may be caused by Δ_1. For parallel change operations, such a condition cannot be drawn from the change log.

While change logs provide a solid basis for analyzing conducted change operations and their consequences, they may not always be available. Concept drift [12,13] focuses on analyzing process execution logs to detect any executed changes on the basic schema. By analyzing the activities present in the change log over a long period of time, this approach can estimate if and which changes have been applied to a process schema.

7 Conclusion and Future Work

In this paper we have discussed how process change operation similarity measures, which calculate the similarity of two process change operations, can be extended to sets of change sequences, i.e. change logs. Change trees serve as a basis, since they provide a compact representation of change sequences. An analysis of the similarity of change sequences can provide interesting insights for many situations in different application domains: In the nursing home, for example, nurses can compare sets of therapies which have been applied to different patients over a long time. Using different similarity measures, different features of the change sequences can be analyzed, e.g. the diversity of required personnel, resources or time. The presented approach can be used with any similarity measure which returns a numeric value within a certain range; thus, it is very flexible and extensible. In a practical setting, the choice of the similarity metric has a huge impact on the information being displayed in a change tree. Thus, future work will analyze the impact of using different similarity notions and aggregation strategies in practical settings.

References

1. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30409-5
2. Günther, C., Rinderle-Ma, S., Reichert, M., van der Aalst, W.: Using process mining to learn from process changes in evolutionary systems. Int. J. Bus. Process Integr. Manage. **3**, 61–78 (2008)
3. Kaes, G., Rinderle-Ma, S.: Mining and querying process change information based on change trees. In: Barros, A., Grigori, D., Narendra, N.C., Dam, H.K. (eds.) ICSOC 2015. LNCS, vol. 9435, pp. 269–284. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48616-0_17
4. Kaes, G., Rinderle-Ma, S.: On the similarity of process change operations. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 348–363. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_22
5. Allen, F.E.: Control flow analysis. ACM Sigplan Not. **5**, 1–19 (1970)
6. Ingvaldsen, J.E., Gulla, J.A.: Industrial application of semantic process mining. Enterp. Inf. Syst. **6**, 139–163 (2012)

7. Rinderle-Ma, S., Reichert, M., Jurisch, M.: On utilizing web service equivalence for supporting the composition life cycle. Int. J. Web Serv. Res. **8**, 41–67 (2011)
8. Bron, C., Kerbosch, J.: Algorithm 457: finding all cliques of an undirected graph. Commun. ACM **16**, 575–577 (1973)
9. Becker, M., Laue, R.: A comparative survey of business process similarity measures. Comput. Ind. **63**, 148–167 (2012)
10. Van Dongen, B., Dijkman, R., Mendling, J.: Measuring similarity between business process models. In: Bellahsène, Z., Léonard, M. (eds.) CAiSE 2008. LNCS, vol. 5074, pp. 450–464. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-69534-9_34
11. Pflug, J., Rinderle-Ma, S.: Process instance similarity: potentials, metrics, applications. In: Debruyne, C., et al. (eds.) OTM 2016. LNCS, vol. 10033, pp. 136–154. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_8
12. Bose, R.P.J.C., van der Aalst, W.M.P., Žliobaitė, I., Pechenizkiy, M.: Handling concept drift in process mining. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 391–405. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_30
13. Carmona, J., Gavaldà, R.: Online techniques for dealing with concept drift in process mining. In: Hollmén, J., Klawonn, F., Tucker, A. (eds.) IDA 2012. LNCS, vol. 7619, pp. 90–102. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34156-4_10

Evaluation of Business Process Modeling Techniques (BPMDS 2018)

An Experimental Evaluation of the Generalizing Capabilities of Process Discovery Techniques and Black-Box Sequence Models

Niek Tax[1](✉), Sebastiaan J. van Zelst[1], and Irene Teinemaa[2]

[1] Eindhoven University of Technology, Eindhoven, The Netherlands
{n.tax,s.j.v.zelst}@tue.nl
[2] University of Tartu, Tartu, Estonia
irene.teinemaa@ut.ee

Abstract. A plethora of *automated process discovery* techniques have been developed which aim to discover a process model based on event data originating from the execution of business processes. The aim of the discovered process models is to describe the control-flow of the underlying business process. At the same time, a variety of *sequence modeling* techniques have been developed in the machine learning domain, which aim at finding an *accurate*, not necessarily *interpretable*, model describing sequence data. Both approaches ultimately aim to find a model that *generalizes* the behavior observed, i.e., they describe behavior that is likely to be part of the underlying distribution, whilst disallowing unlikely behavior. While the generalizing capabilities of process discovery algorithms have been studied before, a comparison, in terms of generalization, w.r.t. sequence models is not yet explored. In this paper we present an experimental evaluation of the generalizing capabilities of automated process discovery techniques and black-box sequence models, on the basis of *next activity prediction.* We compare a range of process discovery and sequence modeling techniques on a range of real-life datasets from the business process management domain. Our results indicate that LSTM neural networks more accurately describe previously unseen traces (i.e., test traces) than existing process discovery methods.

Keywords: Process mining · Behavioral generalization
Next activity prediction · Process discovery · Sequence modeling

1 Introduction

In recent years, the spectacular off-take of data-driven business process analysis, i.e., *process mining* [1], has lead to the availability of a wide variety of tools and techniques that allow business owners to get a better understanding of the execution of their processes. The vast majority of existing process mining techniques either focuses on *process discovery*, i.e., discovering a process model based on process execution data, or *conformance checking*, i.e., assessing whether a process model indeed is in correspondence with the observed behavior. Process discovery techniques allow a business analyst to get a static view of the process, aiding in restructuring and/or reorganization of the process. In recent years, many algorithms have been developed that automatically discover a model of the process from execution data [3,7,16,22,23,25,37,38]. These algorithms' resulting process models provide a visual and interpretable model of the process. The main challenge of process discovery is to accurately generalize the behavior to unseen but possible executions of the business process.

The data on which process discovery algorithms typically operate, i.e. *event logs*, are typically formalized as a collection of sequences of business process activities, i.e. *traces*. Within machine learning, techniques have been developed for *sequence modeling*, addressing the task of finding a model that accurately describes a collection of sequences. Well-known examples of sequence models include n-gram models [14], Markov models, and Recurrent Neural Networks (RNNs) [18]. In contrast to process discovery techniques, sequence modeling techniques are less domain specific, e.g., they do not explicitly account for the parallel behavior that is often inherently present in business processes. Sequence models have been successfully applied to sequence data in many other application domains, including natural language modeling [14], music sequences [24], and DNA sequences in bioinformatics [33]. Moreover, where process discovery aims to find interpretable models, sequence models are generally *black-box models*, focusing purely on accurately describing the sequences.

While the fields of process discovery and sequence modeling are conceptually very close, the main difference between the two approaches is related to the visual interpretability of the result as well as the explicit assumption on the existence of parallelism in the input data. Generally, interpretability of discovered models is seen as important in the process mining/BPM field. However, we argue that this is not always the case and that in some circumstances the accuracy and generalizing capabilities of the model are more important than its interpretability. In fact, recent efforts in the area of *predictive business process monitoring*, i.e., the prediction of expected properties of process instances at runtime (e.g., the outcome, remaining cycle time) have already applied black-box sequence models in the BPM domain. For example, the recent interest in deep learning [21] motivated several researchers to study the applicability of neural networks for the purpose of predictive business process monitoring [15,28,34,35]. However, to date, it has not been investigated how the capabilities of sequence models to describe the control-flow of a process compares to existing process discovery techniques.

In this paper, we investigate the applicability of both process discovery techniques and black-box sequence models on *next activity prediction*. In particular, we assess to what degree the techniques are able to generalize, i.e. to account for unseen behavior. We additionally present means to use discovered process models as probabilistic classifiers, i.e., to predict a probability distribution over

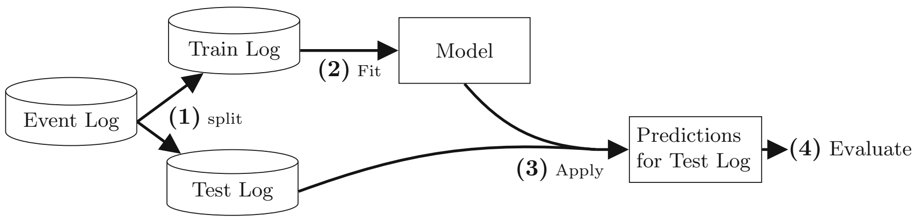


Fig. 1. Overview of the proposed evaluation.

the next activity that follows some prefix of a case. This enables the comparison of existing process discovery and sequence modeling techniques, using standard evaluation methods from the sequence modeling domain. Figure 1 shows the setup for this comparison, where an event log is first split into a train and a test log. A model (i.e., either a sequence model or a process model) is fitted on the training log, which is then used to generate probability distributions over the next activity for each prefix of the test log. The quality of the generated probability distributions for the prefixes in the test log indicates the degree to which the model is able to generalize to unseen data. We compare sequence models and process discovery techniques on a range of real-life event logs from the BPM field.

The remainder of this paper is structured as follows. In Sect. 2, we describe related work. In Sect. 3, we describe basic concepts and notations that are used throughout the paper. In Sect. 4 we describe how to use Petri nets as probabilistic classifiers to predict the next activity given a prefix, and we give an overview of sequence modeling techniques. We describe the experimental setup in Sect. 5 and discuss the results of the experiments in Sect. 6. Finally, we conclude this paper and identify several interesting areas of future work in Sect. 7.

2 Related Work

Several approaches have been proposed in the literature to tackle the challenge of predicting the next activity in an ongoing business process instance. Some of these methods encode the available data as a feature vector and apply a classifier to predict the next event [27,31,36]. Others discover a process/sequence model from the control flow using sequential pattern mining [8], Markov models [20] or a Probabilistic Finite Automaton [4]. Often, as a second step after discovering the process model, classifiers are built for each state in the model, enabling to include the data payload of the ongoing case into the prediction process [8,20]. More recently, deep learning approaches (mostly LSTMs) have been applied to the task of next activity prediction [15,28,34]. However, none of these studies compare their method to existing process discovery techniques. In this work we aim to bridge this gap by comparing the most widely used representatives from the sequence modeling field to well-known process discovery techniques. Furthermore, while most of the next activity prediction methods strive for a high accuracy for a given process instance, our focus in this paper is on assessing the generalizing capabilities of the sequence modeling/process discovery techniques in terms of control-flow, i.e. we tackle the question of how good are the states that each technique is able to learn.

Other work in the predictive monitoring field focuses on predicting the entire continuation of the case beyond the immediate next activity [32,34], deadline violations [29], remaining cycle time [34], and the outcome of a case [11,35].

3 Background

In this section, we introduce concepts used in later sections of this paper.

3.1 Events, Sequences, and Sequence Databases

X^* denotes the set of all sequences over a set X and $\sigma = \langle a_1, a_2, \dots, a_n \rangle$ a sequence of length n, with $\sigma(i) = a_i$ and $|\sigma| = n$. $\langle\rangle$ is the empty sequence and $\sigma_1 \cdot \sigma_2$ is the concatenation of sequences σ_1 and σ_2. $hd^k(\sigma) = \langle a_1, a_2, \dots, a_k \rangle$ is the prefix of length k (with $0 < k < |\sigma|$) of sequence σ, for example, $hd^2(\langle a, b, c, d, e \rangle) = \langle a, b \rangle$. A multiset (or bag) over X is a function $B : X \to \mathbb{N}$ which we write as $[a_1^{w_1}, a_2^{w_2}, \dots, a_n^{w_n}]$, where for $1 \leq i \leq n$ we have $a_i \in X$ and $w_i \in \mathbb{N}^+$. The set of all multisets over X is denoted $\mathcal{B}(X)$. A partial function $f \colon X \nrightarrow Y$ is a function which is not necessarily defined on each element in X. A partial function $f \in X \nrightarrow Y$, can be lifted to sequences over X using: (1) $f(\langle\rangle) = \langle\rangle$; (2) for any $\sigma \in X^*$ and $x \in X$:

$$f(\sigma \cdot \langle x \rangle) = \begin{cases} f(\sigma) \cdot \langle f(x) \rangle & \text{if } f(x) \in Y, \\ f(\sigma) & \text{otherwise.} \end{cases}$$

An *event* e denotes the occurrence of a process activity. We write Σ to denote the universe of business process activities. A *trace* is a sequence $\sigma \in \Sigma^*$. An *event log* is a finite multiset of sequences, $L \in \mathcal{B}(\Sigma^*)$. For example, the event log $L = [\langle a, b, c \rangle^2, \langle b, a, c \rangle^3]$ consists of two occurrences of trace $\langle a, b, c \rangle$ and three occurrences of trace $\langle b, a, c \rangle$.

3.2 Process Models and Process Discovery

We use Petri nets to represent process models, as most state-of-the-art process discovery discover Petri net-like models. A Petri net is a directed bipartite graph consisting of places (depicted as circles) and transitions (depicted as rectangles), connected by arcs. A transition describes an activity, while places represent the enabling conditions of transitions. Labels of transitions indicate the type of activity that they represent. Unlabeled transitions (τ-transitions) represent invisible transitions (depicted as gray rectangles), which are only used for routing purposes and are typically unobservable. As an example of a Petri net, consider Fig. 2, which contains 7 places and 6 transitions. The activity corresponding to transition t_1 is activity A, whereas transition t_3 is unobservable.

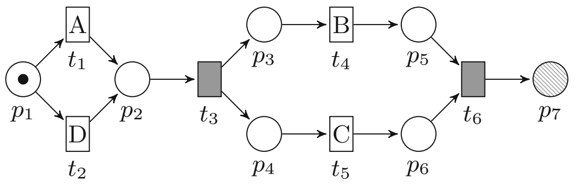


Fig. 2. An example accepting Petri net APN with $\mathfrak{L}(APN) = \{\langle A,B,C\rangle, \langle A,C,B\rangle, \langle D,B,C\rangle, \langle D,C,B\rangle\}$.

Definition 1 (Labeled Petri net). *A* labeled Petri net $N = (P,T,F,\ell)$ *is a tuple where P is a finite set of places, T is a finite set of transitions such that $P \cap T = \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ describes the Petri net flow relation (graphically represented by means of arcs), and $\ell : T \nrightarrow \Sigma$ is a partial labeling function that assigns a label to a transition, or leaves it unlabeled (the τ-transitions).*

We write $\bullet n$ and $n \bullet$ for the input and output nodes of $n \in P \cup T$ (according to F), e.g. in Fig. 2 we have $p\bullet = \{t_1, t_2\}$ and $\bullet t_3 = \{p_2\}$. If the labeling function is undefined for $t \in T$, we write $\ell(t) = \tau$. A state of a Petri net is defined by its *marking* $m \in \mathcal{B}(P)$ being a multiset of places. A marking is graphically denoted by putting $m(p)$ tokens on each place $p \in P$. For example, consider the marking of the example Petri net in Fig. 2, i.e. $[p_1]$, represented by the black dot drawn in p_1. State changes of a Petri net occur through *transition firings.* A transition t is enabled (can fire) in a given marking m if each input place $p \in \bullet t$ contains at least one token. Once t fires, one token is removed from each input place $p \in \bullet t$ and one token is added to each output place $p' \in t\bullet$, leading to a new marking $m' = m - \bullet t + t \bullet$. Firing a transition t in marking m, yielding marking m', is denoted as step $m \xrightarrow{t} m'$. Several subsequent firing steps are lifted to sequences of firing enabled transitions, written $m \xrightarrow{\gamma} m'$ and $\gamma \in T^*$ is a *firing sequence.*

Defining an *initial* and *final* markings allows to define the *language* accepted by a Petri net as a set of finite sequences of activities. To this end we define the notion of an *accepting Petri net*, i.e. a Petri net including an explicit initial and final marking.

Definition 2 (Accepting Petri Net). *An* accepting Petri net *is a triplet $APN = (N, m_0, m_f)$, where $N = (P,T,F,\ell)$ is a labeled Petri net, $m_0 \in \mathcal{B}(P)$ is its initial marking, and $m_f \in \mathcal{B}(P)$ its final marking. A sequence $\sigma \in \Sigma^*$ is a* trace *of an accepting Petri net APN if there exists a firing sequence $m_0 \xrightarrow{\gamma} m_f$, $\gamma \in T^*$ and $\ell(\gamma) = \sigma$.*

In this paper, places that belong to the initial marking contain a token and places belonging to the final marking are marked as ▨.

The *language* $\mathfrak{L}(APN)$ of an accepting Petri net is defined as the set of all its traces, i.e., $\mathfrak{L}(APN) = \{\sigma \in \Sigma^* \mid \exists \gamma \in T^*(\ell(\gamma) = \sigma \wedge m_0 \xrightarrow{\gamma} m_f)\}$, which is potentially of infinite size, i.e. when APN contains loops. Figure 2 shows an example of an accepting Petri net with $\mathfrak{L}(APN) = \{\langle A,B,C\rangle,$

$\langle A, C, B\rangle, \langle D, B, C\rangle, \langle D, C, B\rangle\}$. While we define the language for accepting Petri nets, in theory, $\mathfrak{L}(M)$ can be defined for any process model M with formal semantics. We denote the universe of process models as $\mathcal{M}$ and assume that for each $M \in \mathcal{M}$, $\mathfrak{L}(M) \subseteq \Sigma^*$ is defined.

A process discovery method is a function $PD : \mathcal{B}(\Sigma^*) \rightarrow \mathcal{M}$ that produces a process model from an event log. The discovered process model should cover as much as possible the behavior observed in the event log (a property called *fitness*) while it should not allow for too much behavior that is not observed in the event log (called *precision*). For an event log L, $\tilde{L} = \{\sigma \in \Sigma^* | L(\sigma) > 0\}$ is the *trace set* of L. For example, for event log $L = [\langle a, b, c\rangle^2, \langle b, a, c\rangle^3]$, $\tilde{L} = \{\langle a, b, c\rangle\langle b, a, c\rangle\}$. For an event log L and a process model M, we say that L is *fitting* on M if $\tilde{L} \subseteq \mathfrak{L}(M)$. *Precision* is related to the behavior that is allowed by a model M that was not observed in event log L, i.e., $\mathfrak{L}(M) \backslash \tilde{L}$.

4 Next Activity Prediction

In this section, we present several means to predict the probability distribution over the next activity following a given prefix of a case. We first introduce a method based on (discovered) Petri nets in Sect. 4.1 after which we briefly describe sequence models from the machine learning domain in Sect. 4.2.

4.1 Petri Net Based Prediction

To generate a probability distribution over the set of process activities Σ for the event following the prefix σ of a trace using an accepting Petri net APN, we use a two-step approach:

1. We compute the marking m corresponding to prefix σ in Petri net APN.
2. Based on this derived marking m we consider the subset of the process activities Σ that are reachable from m in APN to have nonzero probability and construct a probability distribution over them.

We now continue by detailing these two steps.

Step (1) Computing Prefix-Based Markings in a Petri Net. To deduce what activities follow a prefix σ, using Petri net APN as a sequence model, we need to obtain the marking corresponding to σ. A naive approach to this problem is to play the so-called "token-game", to find a firing sequence $\gamma \in T^*$ that projected on ℓ equals σ, and marks some arbitrary marking m' in APN. Subsequently, based on marking m', we determine the probable next activities.

Finding such sequence is however far from trivial. Since ℓ is a partial function, it is possible that multiple transitions have the same label and that some other transitions have no label at all. Hence, finding such $\gamma \in T^*$ is actually a combinatorial problem. Aside from these model-based issues, in real-life scenarios, noise is possibly present in the event data as well and the event data might

Activity	A	$\gg$	B
Transition	t_1	t_3	t_4
$\ell(t)$	A	τ	B

Activity	A	D	$\gg$	B
Transition	t_1	$\gg$	t_3	t_4
$\ell(t)$	A	$\gg$	τ	B

Fig. 3. Example prefix-alignments for prefixes $\langle A, B\rangle$ and $\langle A, D, B\rangle$ with the example APN of Fig. 2.

not be completely fitting on the discovered process model APN. For these reasons, a prefix is potentially not completely explainable in terms of the underlying model. Therefore, we apply a technique called *prefix-alignments* [2] to find the path through the model that most closely resembles the prefix.

A prefix-alignment allows us to explain a partial sequence of observed behavior in context of an accepting Petri net. In particular, such alignment assumes that the behavior starts from the initial marking of the model, yet is incomplete, i.e. future behavior is potentially possible. Essentially, a prefix-alignment maps each activity in a prefix to the execution of a transition in the model. Observe that such mapping manipulates the marking of the corresponding model. If we are able to construct a direct mapping between an observed activity and the execution of a transition, we call such mapping a *synchronous move*. In some cases, we observe behavior in the prefix that we are not able to explain in terms of the model, in such case, we decided not to map an activity on the execution of a transition. Such construct is called a *log move*. The reverse is also possible, i.e. we observe behavior that is likely according to the model, however, we did not observe an activity that actually enables this behavior. In such case we insert the firing of a transition without the explicit binding to an observed activity, which we call a *model move*.

Consider Fig. 3 in which we depict two prefix-alignments of two different prefixes of traces, i.e. $\langle A, B\rangle$ and $\langle A, D, B\rangle$, originating from the process as described by the example accepting Petri net in Fig. 2. The leftmost prefix-alignment refers to a behavioral prefix that fits completely with the model, i.e. we map the A activity in the trace onto an execution of t_1, for which we have $\ell(t_1) = A$, i.e. a synchronous move. Subsequently we observe a model move on t_3, with $\ell(t_3) = \tau$, i.e. inherently unobservable, and finally we map the second activity, i.e. B, on the execution of t_4. The prefix-alignment of prefix $\langle A, D, B\rangle$ depicts some deficiencies. We map the first activity, i.e. A, on the execution of transition t_1. The second activity, i.e. D is not mapped on the execution of any transition, i.e. there is a choice between A and D in the model, and hence we obtain a log move. Again, we use a model move on t_3 to finally obtain a synchronous move on activity B.

In general, a multitude of prefix-alignments exists for a given prefix and a process model. For example, in trace $\langle A, D, B\rangle$, as opposed to the right-most alignment in Fig. 3, we are also able to synchronize on D rather than A. The only requirement is that, after explaining the prefix in terms of the model, the corresponding marking still allows us to reach the final marking, in some way. Often we therefore assign costs to the different types of moves as presented earlier.

Typically, synchronous moves have zero-costs, model and log moves usually get a cost of 1 assigned[1]. Computing prefix-alignments is performed by means of solving a shortest path problem on an extended state-space of the accepting Petri net [2]. The costs of a move define the edge costs within such state-space.

For the purpose of this paper, we simply assume the fact that given a prefix and a process model, we are able to obtain the corresponding marking in the accepting Petri net. Observe that, as described, prefix-alignments are the most suitable approach. However, in general any technique that given a prefix and a model results in a corresponding marking is feasible to use.

Step (2) Generating Next Activity Distributions. We propose two ways to generate a probability distribution describing the next activity, based on a marking m obtained for a prefix σ in step 1. From a certain marking, several transitions can be enabled at the same time, e.g. consider marking $[p_1]$ in Fig. 2, in which both transition t_1 and t_2 are enabled. Note that the same holds for marking $[p_3, p_4]$ in which both transition t_4 and t_5 are enabled. Both in the case of choice and parallel constructs, multiple transitions are usually enabled. Alternatively, e.g. in marking $[p_2]$, the only transition that is enabled, which is t_3, has no associated label. For a given marking $m \in \mathcal{B}(P)$ in an accepting Petri net APN, $\omega(m) = \{t | t \in T \wedge \bullet t \subseteq m\}$ is the set of enabled, or firable, transitions. Note that $\omega(m)$ can contain labeled as well as invisible transitions. A probability mass function $prob_m : T \rightarrow [0, 1]$ assigns a firing probability to each transition that is enabled from marking m, such that $\Sigma_{t \in \omega(m)} prob_m(t) = 1$. In the first approach, we assume this probability distribution over the enabled transitions to be a uniform categorical distribution, i.e., $prob_{m(t)}^{uniform} = \begin{cases} \frac{1}{|\omega(m)|} & \text{if } \; t \in \omega(m), \\ 0 & \text{otherwise.} \end{cases}$

To transform the probability distribution $prob_m$ over the next transition to fire into a probability distribution over the next activity, we apply the following procedure. Starting from marking m in Petri net APN we pick an enabled transition at random according to probability distribution $prob_m^{uniform}$. Whenever that transition has a corresponding label, we pick it as the next activity. If it relates to an unobservable transition, we fire it, leading to a new marking m' and apply the same procedure from up-until we select a transition that has a label. This procedure to select the next activity from a marking m is repeated a number of times and we set the probabilities over the activities according to how often they were selected, i.e. by Monte Carlo simulation. For example, in marking p_2 in Fig. 2 we always fire transition t_3 which yields marking $[p_3, p_4]$. In that marking we randomly pick any of the two transitions, i.e. t_4 or t_5, and register their corresponding labels as a next activity, i.e. either B or C. Thus, we obtain $prob_{[p_2]}^{uniform}(B) = prob_{[p_2]}^{uniform}(C) = 0.5$.

In the second approach, after discovering a process model based on the training log, we compute an empirical distribution $prob_m^{empirical}$. This is rather

[1] Except for model moves that relate to unobservable activities, which also get cost 0 assigned.

straightforward: for each prefix of each trace in the training log we apply step 1 to obtain the corresponding marking m in the discovered Petri net APN, and we base $prob_m^{empirical}$ on how often each enabled transition $t \in \omega(m)$ was fired when this marking was reached in the training log. This leads to a probability mass function for each marking in the model that is trained/estimated based on the training data. We again apply the same Monte Carlo sampling approach to transform $prob_m^{empirical}$ into a probability distribution over the next activity, instead of over the next transition. For example, in marking p_2 in Fig. 2 we always fire transition t_3 which yields marking $[p_3, p_4]$. From that marking, both B and C are enabled and probability mass values $prob_{[p_2]}^{emperical}(B)$ and $prob_{[p_2]}^{emperical}(C)$ are proportional to how often activities B and C followed marking p_2 in the training log.

Both the Petri-net-based probabilistic classifier based on $prob_m^{uniform}$ and based on the trained $prob_m^{empirical}$ have been implemented and are openly available as part of the ProM process mining toolkit [13] in the package *SequencePredictionWithPetriNets*[2].

4.2 Black-Box Sequence Models

We proceed by introducing sequence models from the machine learning field.

Neural Networks and Recurrent Neural Networks. A neural network consists of one layer of *input units*, one layer of output units, and in-between are multiple layers that are referred to as *hidden units*. The outputs of the input units form the inputs for the units of the first *hidden layer* (i.e., the first layer of hidden units), and the outputs of the units of each hidden layer form the input for each subsequent hidden layer. The outputs of the last hidden layer form the input for the output layer. The output of each unit is a function over the weighted sum of its inputs. The weights of this weighted sum performed in each unit are learned through gradient-based optimization from training data that consists of example inputs and desired outputs for those example inputs. Recurrent Neural Networks (RNNs) are a special type of neural networks where the connections between neurons form a directed cycle.

RNNs can be unfolded, as shown in Fig. 4. Each step in the unfolding is referred to as a time step, where x_t is the input at time step t. RNNs can take an arbitrary length sequence as input, by providing the RNN a feature representation of one element of the sequence at each time step. s_t is the hidden state at time step t and contains information extracted from all time steps up to t. The hidden state s is updated with information of the new input x_t after each time step: $s_t = f(Ux_t + Ws_{t-1})$, where U and W are vectors of weights over the new inputs and the hidden state respectively. In practice, either the hyperbolic tangent or the logistic function is generally used for function f, which is referred to as the activation function. The logistic function is defined as: $sigmoid(x) = \frac{1}{1+exp(-x)}$.

[2] https://svn.win.tue.nl/repos/prom/Packages/SequencePredictionWithPetriNets/.

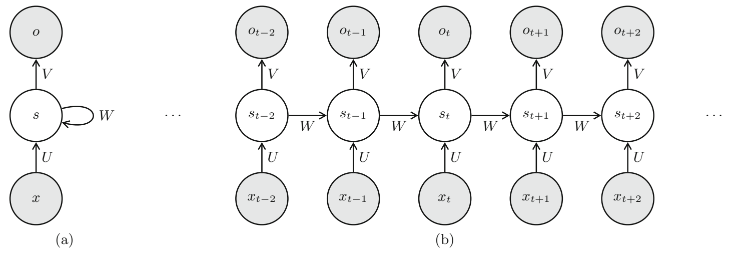


Fig. 4. *(a)* A simple recurrent neural network consisting of a single hidden layer, and *(b)* the recurrent neural network unfolded over time.

In neural network literature, the sigmoid function is often represented by σ, however, to avoid confusion with traces, we fully write *sigmoid*. o_t is the output at step t.

Long Short-Term Memory for Sequence Modeling. A Long Short-Term Memory (LSTM) model [17] is a special Recurrent Neural Network architecture that has powerful modeling capabilities for long-term dependencies. The main distinction between a regular RNN and an LSTM is that the latter has a more complex memory cell C_t replacing s_t. Where the value of state s_t in an RNN is the result of a function over the weighted average over s_{t-1} and x_t, the LSTM state C_t is accessed, written, and cleared through controlling gates, respectively o_t, i_t, and f_t. Information on a new input will be accumulated to the memory cell if i_t is activated. Additionally, the previous memory cell value C_{t-1} can be "forgotten" if f_t is activated. The information of C_t will be propagated to the output h_t based on the activation of output gate o_t. Combined, the LSTM model can be described by the following formulas:

$$\begin{aligned} f_t &= sigmoid(W_f \cdot [h_{t-1}, x_t] + b_f) & i_t &= sigmoid(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_c \cdot [h_{t-1}, x_t] + b_C) & C_t &= f_t * C_{t-1} + i_i * \tilde{C}_t \\ o_t &= sigmoid(W_o[h_{t-1}, x_t] + b_o) & h_t &= o_t * tanh(C_t) \end{aligned}$$

In these formulas all W variables are weights and b variables are biases and both are learned during the training phase.

Gated Recurrent Units. Gated Recurrent Units (GRU) were proposed by Cho et al. [9] as a simpler alternative to the LSTM architecture. In comparison to LSTMs, GRUs do not keep a separate memory cell and instead merge the cell state C_t and hidden state h_t. Furthermore, a GRU combines the input gate i_t and the forget gate f_t into a single *update gate*. While the LSTMs and GRUs are identical in the class of functions that they can learn, GRUs are simpler in the sense that they have fewer model parameters. Empirically, GRUs have been found to outperform LSTMs on several sequence prediction tasks [10].

5 Experimental Setup

In process mining, *generalization*, is commonly described as "the likelihood that the process model is able to describe yet unseen behavior of the observed system" [7]. Several measures have been proposed to measure the generalization of a process model [6,12], all of which calculate the generalization of the process model *with respect to the same event log from which the process model was discovered*. In contrast, in the sequence modeling field it is common to measure generalization by splitting the data into a separate *training* set that is used to learn the model and a *test* set on which it is evaluated how well the model fits this data. Because the test set is a separate data set from the training set, the fit between model and test set can be considered to measure the *generalization* of the model to the test data. Figure 1 describes the experimental setup on a high level. We apply this experimental setup to a collection of event logs and for a collection of discovery/sequence modeling techniques. For each combination of modeling method and event log we make a random 70/30%-split of the log into training log and test log and after generating the model on the training log we evaluate how well the actual next activity predicted for each prefix in the test log fits the probability distribution over all possible next activities according to the model. For each combination of event log and modeling technique we repeat the experiment three times and calculate the 95% confidence interval around the model performance, to prevent that the results are too dependent on the random sampling of the event log into train and test split. We calculate the Brier score [5] between the probability distribution over the next activity for given a prefix and the actual next activity to express the quality of the prediction. The Brier score is a well-known measure to evaluate a probabilistic classifier and it can intuitively be interpreted as being the mean squared error of the predicted likelihood of a given class (i.e. activity), averaged over all activities and over all prefixes in the test set.

Modeling Methods. As modeling techniques we apply the RNN, LSTM, GRU neural networks that are described in Sect. 4.2. For each of those types of neural networks we explore its performance using only one layer as well as using three layers. We learn the weights of the model with Adam [19]. Furthermore, we first order as well as second order Markov models, i.e. probabilistic models where the state of the model depends on the last k events, with k the order of the Markov model. The predicted probability distribution over the next activity is static for each state. We explore higher order Markov models by including a sequence model called *all k-order Markov models* (AKOM) [30] that fits all Markov models of any order k to the training set. When making an prediction (on the test set), AKOM uses the Markov model with the highest k that has a state that matches the test sequence.

We evaluate the following process discovery techniques: Inductive Miner (IM) [22], the Inductive Miner with infrequency filtering (IMf) [23] (using different thresholds), the Heuristics Miner [37], the Split Miner [3], the Hybrid-ILP

Miner [38], and the Evolutionary Tree Miner (ETMd) [7]. Unless parameter values of these process discovery techniques are stated explicitly they are left to their default values. We add the flower model, i.e. the model that allows for all behavior over the set of activities, for comparison. Note that the Brier score of the flower model with a uniform distribution represents random guessing.

Event Logs. We evaluate the generalizing capabilities of process discovery techniques and sequence modeling techniques on three real-life event logs. The first log is the receipt phase log from the WABO project[3], containing 8577 events of 27 activities originating from 1434 cases of the receipt phase of the building permit application process at a Dutch municipality. The second log contains cases from a financial loan application process at a large financial institute[4], consisting of 164506 events divided over 13087 cases and 23 activities. As the third log we use the sepsis event log [26], containing medical care pathways of 1050 sepsis patients, for which in total 15214 events were logged from 16 different activities.

6 Results

Table 1 shows the results in terms of Brier score on the three datasets for each of the techniques. The worst Brier score value of each column in the table is colored red and the best value is colored green, with the other values taking a color in between. For the process discovery methods it seems that learning the probability distribution on the training data leads to better fitting probability distributions on the test data compared to assuming this probability distribution to be uniform. Note, however, that when using the discovered process model for stakeholder communication about the process there are typically no branching probabilities shown in the model. Therefore, one could say that the uniform distribution matches the graphical representation of the Petri net. The Split Miner [3] is the best performing process discovery technique on two of the three logs when we learn the probability distribution per marking from the training data. The Hybrid-ILP Miner [38] is the best performing process discovery technique on two of the three logs when we consider a uniform distribution per marking, indicating that this miner finds and the most informative states (i.e., markings) and communicates the behavior of the process most effectively to the process analyst.

On all three event logs the Brier scores for the neural network models are considerably better, meaning that they provide considerably more accurate probability distributions over the next event for prefixes from previously unseen traces. These models are black-box and therefore not suitable for communicating the process, however, they seem to be much better generalizing the behavior of the process. We found that it depends per log which neural network architecture

[3] https://doi.org/10.4121/uuid:a07386a5-7be3-4367-9535-70bc9e77dbe6.
[4] https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.

Table 1. Experimental results showing the 95% confidence interval lower bound (LB), upper bound (UB) and the mean of the Brier score.

Method	Receipt Phase			BPI'12			SEPSIS		
	LB	UB	μ	LB	UB	μ	LB	UB	μ
Process Discovery: Uniform distribution per marking									
Flower	0.0371	0.0391	0.0381	0.0417	0.0417	0.0417	0.0675	0.0846	0.0761
IM [22]	0.0321	0.0356	0.0338	0.0301	0.0326	0.0314	0.0559	0.0665	0.0612
IMf 20% [23]	0.0209	0.0239	0.0223	0.0292	0.0295	0.0293	0.0475	0.0497	0.0486
IMf 50% [23]	0.0155	0.0228	0.0191	0.0292	0.0295	0.0293	0.0673	0.0844	0.0759
HM [37]	0.0237	0.0252	0.0245	0.0267	0.0272	0.0269	0.0437	0.0448	0.0442
SM [3]	0.0240	0.0285	0.0262	0.0250	0.0255	0.0252	0.0548	0.0600	0.0574
Hybrid-ILP [38]	0.0155	0.0179	0.0167	0.0227	0.0300	0.0263	0.0413	0.0439	0.0426
ETMd [7]	0.0168	0.0224	0.0196	0.0259	0.0315	0.0287	0.0513	0.0539	0.0526
Process Discovery: Trained distribution per marking									
Flower	0.0327	0.0345	0.0336	0.0402	0.0402	0.0402	0.0540	0.0543	0.0542
IM [22]	0.0228	0.0282	0.0255	0.0276	0.0297	0.0287	0.0420	0.0490	0.0455
IMf 20% [23]	0.0209	0.0239	0.0224	0.0228	0.0255	0.0241	0.0381	0.0409	0.0395
IMf 50% [23]	0.0144	0.0162	0.0153	0.0228	0.0255	0.0241	0.0532	0.0535	0.0534
HM [37]	0.0174	0.0187	0.0181	0.0228	0.0234	0.0231	0.0359	0.0385	0.0372
SM [3]	0.0092	0.0105	0.0099	0.0225	0.0227	0.0226	0.0387	0.0439	0.0413
Hybrid-ILP [38]	0.0155	0.0179	0.0167	0.0286	0.0404	0.0345	0.0399	0.0425	0.0412
ETMd [7]	0.0104	0.0124	0.0114	0.0254	0.0272	0.0263	0.0389	0.0402	0.0396
Neural Network Methods									
RNN (1 layer)	0.0065	0.0080	0.0073	0.0124	0.0127	0.0126	0.0277	0.0282	0.0279
RNN (3 layers)	0.0071	0.0079	0.0075	0.0125	0.0131	0.0128	0.0261	0.0286	0.0273
LSTM (1 layer)	0.0070	0.0075	0.0073	0.0120	0.0123	0.0122	0.0272	0.0273	0.0273
LSTM (3 layers)	0.0066	0.0079	0.0072	0.0123	0.0132	0.0128	0.0283	0.0323	0.0303
GRU (1 layer)	0.0070	0.0081	0.0076	0.0124	0.0126	0.0125	0.0269	0.0280	0.0274
GRU (3 layers)	0.0069	0.0077	0.0073	0.0126	0.0138	0.0132	0.0272	0.0281	0.0276
Markov Model Methods									
1st-order Markov Model	0.0223	0.0245	0.0235	0.0317	0.0318	0.0318	0.0488	0.0493	0.0491
2nd-order Markov Model	0.0216	0.0235	0.0225	0.0256	0.0258	0.0257	0.0468	0.0472	0.0470
AKOM [30]	0.0177	0.0192	0.0185	0.0241	0.0243	0.0242	0.0427	0.0432	0.0429

performs the best, although the differences between their Brier scores are only minor. Comparing process discovery techniques to Markov models, we find that discovered process models are better at generalizing the behavior when we learn a probability distribution per marking. This holds for 1st-order and 2nd-order Markov models as well as AKOM [30].

7 Conclusions and Future Work

In this paper we have introduced two ways in which (discovered) Petri nets can be used as probabilistic classifiers to predict the next activity for a given prefix of a trace: a uniform distribution approach, which uses only the information that is visually communicated by the graphical representation of the Petri net, and an empirical distribution approach that optimizes a categorical probability distribution per marking using a training log. We have compared how well process models discovered with process discovery techniques are able to generalize by producing accurate probability distributions for a test set of prefixes that were not used to obtain the model. On three real-life business process event logs we have compared process discovery techniques with existing (non-interpretable) sequence modeling techniques from the Machine Learning domain, and have found that machine learning models possess better generalizing properties than discovered process models. This shows that machine learning sequence modeling might be a better choice than process discovery methods to model a business process when interpretability of the model is not a requirement. In future work, we would like to extend our analysis to a larger collection of event logs, process discovery techniques, and sequence modeling methods.

References

1. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Adriansyah, A.: Aligning observed and modeled behavior. Ph.D. thesis, Eindhoven University of Technology (2014)
3. Augusto, A., Conforti, R., Dumas, M., La Rosa, M.: Split miner: discovering accurate and simple business process models from event logs. In: IEEE International Conference on Data Mining, pp. 1–10. IEEE (2017)
4. Breuker, D., Matzner, M., Delfmann, P., Becker, J.: Comprehensible predictive models for business processes. MIS Q. **40**(4), 1009–1034 (2016)
5. Brier, G.W.: Verification of forecasts expressed in terms of probability. Mon. Weather Rev. **78**(1), 1–3 (1950)
6. vanden Broucke, S.K.L.M., De Weerdt, J., Vanthienen, J., Baesens, B.: Determining process model precision and generalization with weighted artificial negative events. IEEE Trans. Knowl. Data Eng. **26**(8), 1877–1889 (2014)
7. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: On the role of fitness, precision, generalization and simplicity in process discovery. In: Meersman, R., et al. (eds.) OTM 2012. LNCS, vol. 7565, pp. 305–322. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33606-5_19
8. Ceci, M., Lanotte, P.F., Fumarola, F., Cavallo, D.P., Malerba, D.: Completion time and next activity prediction of processes using sequential pattern mining. In: Džeroski, S., Panov, P., Kocev, D., Todorovski, L. (eds.) DS 2014. LNCS (LNAI), vol. 8777, pp. 49–61. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11812-3_5
9. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Conference on Empirical Methods in Natural Language Processing. ACL (2014)
10. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: NIPS Deep Learning and Representation Learning Workshop (2014)
11. Di Francescomarino, C., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring. IEEE Trans. Serv. Comput. (2016)

12. van Dongen, B.F., Carmona, J., Chatain, T.: A unified approach for measuring precision and generalization based on anti-alignments. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 39–56. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_3
13. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM framework: a new era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 444–454. Springer, Heidelberg (2005). https://doi.org/10.1007/11494744_25
14. Dunning, T.: Statistical identification of language. Computing Research Laboratory, New Mexico State University (1994)
15. Evermann, J., Rehse, J.R., Fettke, P.: Predicting process behaviour using deep learning. Decis. Support Syst. **100**, 129–140 (2017)
16. Goedertier, S., Martens, D., Vanthienen, J., Baesens, B.: Robust process discovery with artificial negative events. J. Mach. Learn. Res. **10**, 1305–1340 (2009)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
18. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. **79**(8), 2554–2558 (1982)
19. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: International Conference for Learning Representations (2015)
20. Lakshmanan, G.T., Shamsi, D., Doganata, Y.N., Unuvar, M., Khalaf, R.: A markov prediction model for data-driven semi-structured business processes. Knowl. Inf. Syst. **42**(1), 97–126 (2015)
21. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015)
22. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs - a constructive approach. In: Colom, J.-M., Desel, J. (eds.) PETRI NETS 2013. LNCS, vol. 7927, pp. 311–329. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38697-8_17
23. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Lohmann, N., Song, M., Wohed, P. (eds.) BPM 2013. LNBIP, vol. 171, pp. 66–78. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06257-0_6
24. Logan, B., Chu, S.: Music summarization using key phrases. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 2, pp. II749–II752. IEEE (2000)
25. Maggi, F.M., Mooij, A.J., van der Aalst, W.M.P.: User-guided discovery of declarative process models. In: IEEE Symposium on Computational Intelligence and Data Mining, pp. 192–199. IEEE (2011)
26. Mannhardt, F., Blinde, D.: Analyzing the trajectories of patients with sepsis using process mining. In: RADAR+EMISA, vol. 1859, pp. 72–80. CEUR-ws.org (2017)
27. Márquez-Chamorro, A.E., Resinas, M., Ruiz-Cortés, A., Toro, M.: Run-time prediction of business process indicators using evolutionary decision rules. Expert Syst. Appl. **87**, 1–14 (2017)
28. Mehdiyev, N., Evermann, J., Fettke, P.: A multi-stage deep learning approach for business process event prediction. In: IEEE Conference on Business Informatics, vol. 1, pp. 119–128. IEEE (2017)
29. Pika, A., van der Aalst, W.M.P., Fidge, C.J., ter Hofstede, A.H.M., Wynn, M.T.: Predicting deadline transgressions using event logs. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 211–216. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_22

30. Pitkow, J., Pirolli, P.: Mining longest repeating subsequences to predict worldwide web surfing. In: USENIX Symposium on Internet Technologies and Systems, pp. 13–26 (1999)
31. Pravilovic, S., Appice, A., Malerba, D.: Process mining to forecast the future of running cases. In: Appice, A., Ceci, M., Loglisci, C., Manco, G., Masciari, E., Ras, Z.W. (eds.) NFMCP 2013. LNCS (LNAI), vol. 8399, pp. 67–81. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08407-7_5
32. van der Spoel, S., van Keulen, M., Amrit, C.: Process prediction in noisy data sets: a case study in a Dutch hospital. In: Cudre-Mauroux, P., Ceravolo, P., Gašević, D. (eds.) SIMPDA 2012. LNBIP, vol. 162, pp. 60–83. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40919-6_4
33. Stanke, M., Waack, S.: Gene prediction with a hidden Markov model and a new intron submodel. Bioinformatics **19**(suppl. 2), ii215–ii225 (2003)
34. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
35. Teinemaa, I., Dumas, M., Maggi, F.M., Di Francescomarino, C.: Predictive business process monitoring with structured and unstructured data. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 401–417. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_23
36. Unuvar, M., Lakshmanan, G.T., Doganata, Y.N.: Leveraging path information to generate predictions for parallel business processes. Knowl. Inf. Syst. **47**(2), 433–461 (2016)
37. Weijters, A.J.M.M., Ribeiro, J.T.S.: Flexible heuristics miner (FHM). In: IEEE Symposium on Computational Intelligence and Data Mining, pp. 310–317. IEEE (2011)
38. van Zelst, S.J., van Dongen, B.F., vander Aalst, W.M.P., Verbeek, H.M.W.: Discovering workflow nets using integer linear programming. Computing 1–28 (2017)

Towards a Methodology for Case Model Elicitation

Marcin Hewelt(✉), Felix Wolff, Sankalita Mandal, Luise Pufahl, and Mathias Weske

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
{marcin.hewelt,felix.wolff,sankalita.mandal, luise.pufahl,mathias.weske}@hpi.de

Abstract. Case management is increasingly used to capture and enact flexible, knowledge-intensive processes in organizations. None of the existing case management approaches provides a methodology for case model elicitation. In this contribution, two elicitation methods for fragment-based case management are presented: one which focuses on the control-flow view, the *process-first* method, and one which has a data-centric view, the *object lifecycle-first* method. Following the design science process, both methods are evaluated in separate process elicitation workshops. The results indicate that the *object lifecycle-first* yields more flexible and consistent case models. However, the first method might be helpful for scenarios where the view on the main process is needed.

Keywords: Case model · Object lifecycle · t.BPM · Process elicitation

1 Introduction

Case Management has been proposed to support flexible and knowledge-intensive business processes [1] that cannot be represented well using standard approaches like BPMN (Business Process Model and Notation) [2] and traditional process engines. While there has been considerable research work in case management in recent years, e.g. [3,4], there has been a surprising lack of work in case model elicitation, i.e. in the question how case models are designed. Consequently, most approaches proposed so far lack concrete methods that practitioners could apply to design case models. This constitutes a serious threat to the applicability of case management approaches in practice.

To close this gap, we strive towards a methodology[1] for the elicitation of case models. Modeling methods give concrete steps and guidelines how modelers should apply and combine the constructs of a specific modeling language to create a model. As such, they are intimately linked to a concrete modeling language. In this contribution we therefore focus on the Chimera case management approach [5]. We propose two different case model elicitation methods for

[1] Methodology is defined as study of the methods that are applied in a field, here case management.

Chimera, called *process-first* and *object lifecycle-first*. Both methods are based on the common process modeling method by Dumas et al. [6] and the workshop-based tangible BPM (t.BPM) method of Lübbe [7]. While the first method starts with elicitating the process fragments, the second method starts with object lifecycles of the involved data objects, however, both methods result in a case model. We evaluate and compare the methods by conducting an experiment, following the design science methodology [8,9]. Nonetheless, our methodology is generic enough to be adaptable for other fragment-based Case Management approaches.

The remainder of the paper is structured as follows. Section 2 describes related work and motivates our research. An overview of the Chimera approach is given in Sect. 3. The methods for case model elicitation are presented in Sect. 4. The experiment for both methods and its results are described in Sect. 5, which also compares the methods and discusses benefits and limitations of both. Finally, we conclude in Sect. 6.

2 Related Work

Traditional process modeling approaches are not well suited to capture and enact the flexible and knowledge-intensive processes of case workers (see for example [3,4]). On the other hand, Marin et al. (2015) [1] conclude, that case management seems a suitable approach to address the requirements of knowledge-intensive processes. Several case management approaches, e.g. Proclets [10] and PHILharmonic flows [11], propose to split end-to-end process models into smaller fragments that can be combined, resulting in increased flexibility. Those approaches also emphasize the central role of data driving the case. The case models in the Chimera approach for example consist of data classes, their object lifecycles, and a set of process fragments, which are dynamically instantiated and combined at runtime based on data object states [5].

While case modeling approaches themselves are becoming mature, the development of elicitation methods has not kept up. Proclets and PHILharmonic flows, for example, do not provide a method for model elicitation. To our knowledge only the data-centric Business Artifacts approach [12] offers a methodology [13]. However, it is not considered to be a case management approach.

General process modeling approaches like BPMN on the other hand, have a variety of elicitation methods to chose from, e.g. structured interviews [14], workshops [6], or tangible BPM (t.BPM) [7]. A common process modeling method independent from the elicitation techniques is proposed by Dumas et al. [6] which is very general and focuses mainly on process activities. Flexibility aspects are not considered in their method. In addition, process mining techniques [15] can be used to derive process models from collected logs of process executions.

t.BPM has been introduced as a method to elicitate business process models that provides a high degree of user involvement. It is based on the concept of shared ownership of models, so that every process participant gets involved and updates the process model based on his or her experiences. t.BPM stands for tangible business process modeling. It uses a set of tangible shapes that mimic

BPMN activities, gateways, and events. Since t.BPM facilitates a high degree of involvement of process participants and case management stresses on the knowledge workers and the decisions they take, t.BPM is in principle suited to support modeling of cases.

However, case management asks for a different approach, since it is based on different assumptions. In addition, different model elements have to be represented, such as object life cycles of data objects involved in a case. The methods introduced in this paper take into account these requirements and modify t.BPM accordingly.

3 Fragment-Based Case Management

The Chimera approach captures business scenarios in a *case model* that consists of (a) a domain model, (b) a set of object lifecycles, (c) a set of process fragments, and (d) a goal state, also called termination condition. During runtime a case model is instantiated into a *case* that at any time is in a certain *case state*, which changes through knowledge workers performing activities, but also due to external events. Cases are similar to process instances in traditional workflow systems, however, contrary to those, cases can contain several concurrently running fragment instances, as well as data objects.

The *domain model* defines the business objects relevant for the scenario as a set of data classes and their relations. Each data class is a named entity that has a set of attributes, which can assume values from a specified domain. To each data class an *object lifecycle (OLC)* is associated specifying valid behavior of its instances, i.e. data objects (see Fig. 1 for an example). Additionally, OLCs are assigned to the other elements of a case model and govern their behavior. Those latter OLCs for fragments, activities, gateways, events, and the case itself, are generic in the sense that they do not depend on the concrete scenario at-hand. Object lifecycles are state transition systems consisting of states and state transitions, as well as initial and final states.

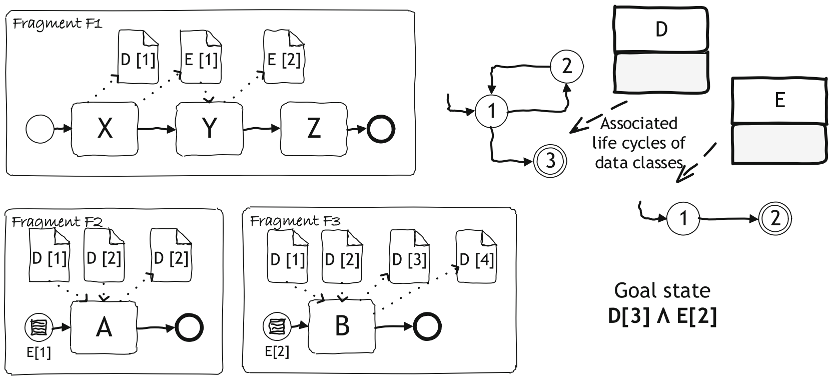


Fig. 1. Visualization of the main concepts of fragment-based case management

Contrary to traditional workflow approaches, the end-to-end process is split into several *process fragments* that describe structured parts of a business scenario. Figure 1 visualizes an abstract example with three fragments. During case enactment they are instantiated and combined based on data object states and external events, thus offering a high level of flexibility. Each fragment, just like a BPMN process model, consists of event, gateway, activity, and data nodes, connected by sequence and data flow arcs. Lifecycles for activities, gateways, and events are based on the ones described in Weske (2012) [16].

To achieve dynamic fragment combination at runtime, it is essential to model data pre- and post-conditions. Data nodes play a central role in this regard, as they specify both the state a data object needs to be in, to be considered a valid input for an activity, as well as the resulting state the activity produces when it terminates. Together, the incoming (resp. outgoing) data nodes of an activity node determine the pre-condition (resp. post-condition) of the activity. An activity might have several input (resp. output) sets, i.e. it can be enabled by (resp. produce) different data objects in different states.

Data state conditions are also used to specify the goal of the case. When this termination condition is fulfilled in the current case state, i.e. the specified data objects are in the specified state, the case worker can terminate the case. Additionally, data state conditions can be used to restrict the instantiation of fragments, as shown in Fig. 1 for fragments F_2 and F_3, which can only be instantiated if data object E is in state 1 and 2, respectively.

4 Methods for Case Model Elicitation

A common process modeling method presented by Dumas et al. [6] consists of the following five steps: (1) Identify the process boundaries, (2) Identify activities and events, (3) Identify resources and their handovers, (4) Identify the control flow, and (5) Identify additional elements (e.g., data objects).

This process modeling method is taken as inspiration and tailored to fragment-based case modeling in our work. Fragment-based case management includes both, the notion of activities and the notion of OLCs. Thus, we provide on the one side a method which focuses on the control-flow view – *process-first* method presented in Sect. 4.1, and on the other side a method which emphasizes the object lifecycle view – *object lifecycle-first* method presented in Sect. 4.2.

4.1 *Process-First* Case Modeling Method

In many business processes, e.g. in health care, flexible behavior can be observed. This means that certain activities may or may not be executed for a case, or they can also be executed at different points in the case. Figure 2 illustrates the design of an *Offer creation process* in which the approval of a customer can be executed at different states of the process. However, the approval does not have to be executed in all cases, only if the customer is new and has not been approved already. In a standard BPMN diagram, this would lead to many permutations and a complex control flow.

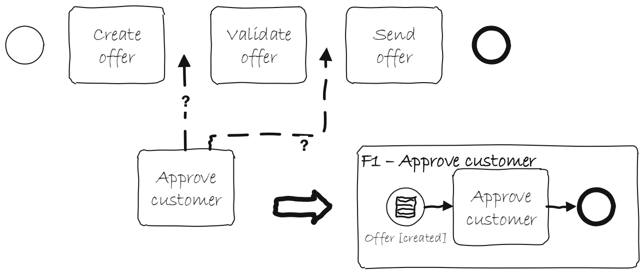


Fig. 2. Design of an *Offer creation process* with flexible behavior: the approval of the customer may or may not be executed at different points in the process

Thus, we propose a method based on the common process modeling which additionally allows to specify flexible behaviour by using the concept of process fragments. This means that a main process fragment is designed, and if flexible behavior is identified, those activities can be outsourced into a new fragment as exemplified in Fig. 2. The enablement of a fragment is described via data pre-conditions (i.e. the start event is a conditional event referencing data object states which have to be fulfilled for starting a fragment). The steps of the *process-first* method are visualized in Fig. 3.

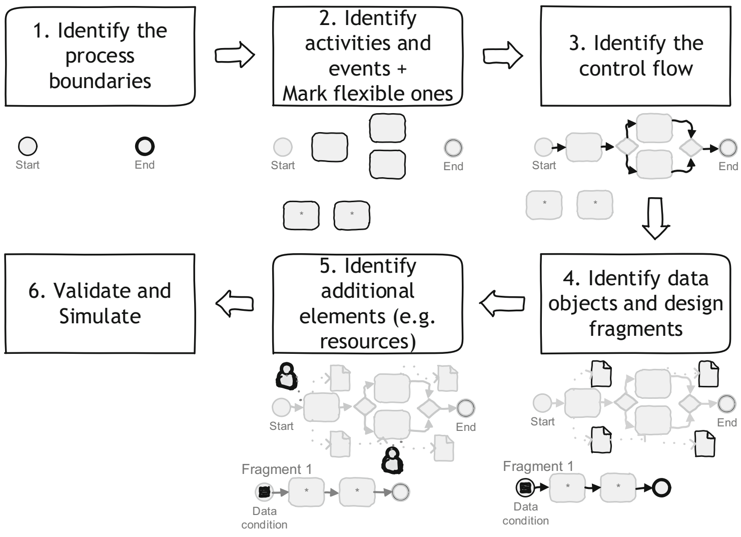

1. Identify the process boundaries
Start
End
2. Identify activities and events + Mark flexible ones
Start
End
3. Identify the control flow
Start
End
4. Identify data objects and design fragments
Start
End
Fragment 1
Data condition
5. Identify additional elements (e.g. resources)
Start
End
Fragment 1
Data condition
6. Validate and Simulate

Fig. 3. Steps of the *Process-first* case modeling method

Similar to the common process modeling method [6], the process boundaries – the start and the end of the main process – have to be identified at first. In the second step, the main activities and events are outlaid in a rough order. If, in this step, activities or events are identified which can occur flexibly during the process execution, those should be marked and outsourced from the main process. Currently, we focus on the main elements of the fragment-based case management approach, namely, process fragments and data objects. Therefore, the resources are handled as additional elements and can be modeled in a later step of this method. In the third step, the control flow between activities and events is defined. If during this step, further activities can be detected which can flexibly occur during process execution, these can also be outsourced. For designing the outsourced fragments, the data start conditions for the fragments have to be identified. In the fourth step, the most important data input and output objects of the process activities are elicited. This means that data notes are added to fragments and connected to activities for visualizing in which state a specific data object is read or written by certain activities. Based on this, fragments with the outsourced activities can be designed. In the fifth step, additional elements, for instance resources, can be added to the process. Inspired by the t.BPM workshop technique, in the last step, the case model should be validated by the process stakeholders. Thereby, it is helpful if the stakeholders simulate the execution of specific cases and check the completeness of the case model.

The result of this method is a main process fragment representing the main process and a set of fragments capturing the flexible behavior, both with their processed data objects. The domain model, the data object lifecycles and the goal state required for a complete case model are indirectly available. By identifying the different types of data nodes depicted in the process fragments, the data classes for the domain model can be identified. Relations between those data classes have to be manually added. The object lifecycles can be derived from the data pre- and post-conditions of the activities. The goal state is the conjunction of the states in which the data objects are at the end event of the main fragment. If several end events exist, then the data object state sets of the end events are connected disjunctively.

4.2 *Object Lifecycle-First* Case Modeling Method

The first method centers around a main process and treats data as second class modeling construct. For certain knowledge-intensive processes, the need for flexibility is higher and is only restricted with regard to the actions that are allowed to be performed on data objects by the activities of the business process. Therefore, a data-centric case modeling method is presented in the following.

For this method, we follow the idea of business artifacts [12] that each business process is "driven" by a main data object which is called the *case object*. By selecting a case object, for example, the *Offer* object for the *Offer creation process*, first its OLC is defined in this method, as shown in Fig. 4. The OLC defines on a high level which actions are possible in a certain case model. These actions can be later detailed by defining a process fragment for each allowed action.

In Fig. 4, the OLC is not visualized as traditional state transition diagram; instead BPMN data nodes are used for representing the data states which are connected with each other for showing possible state transitions. We assume that most stakeholders are more familiar with the BPMN notation. As stated, the

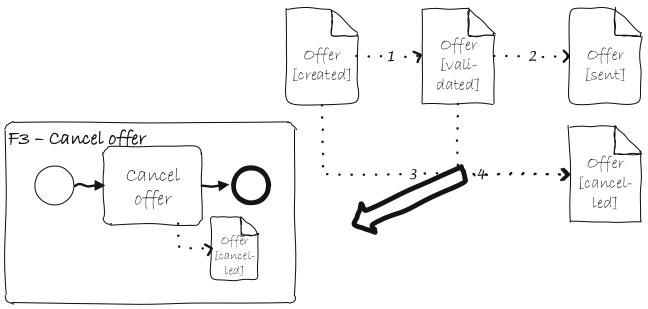
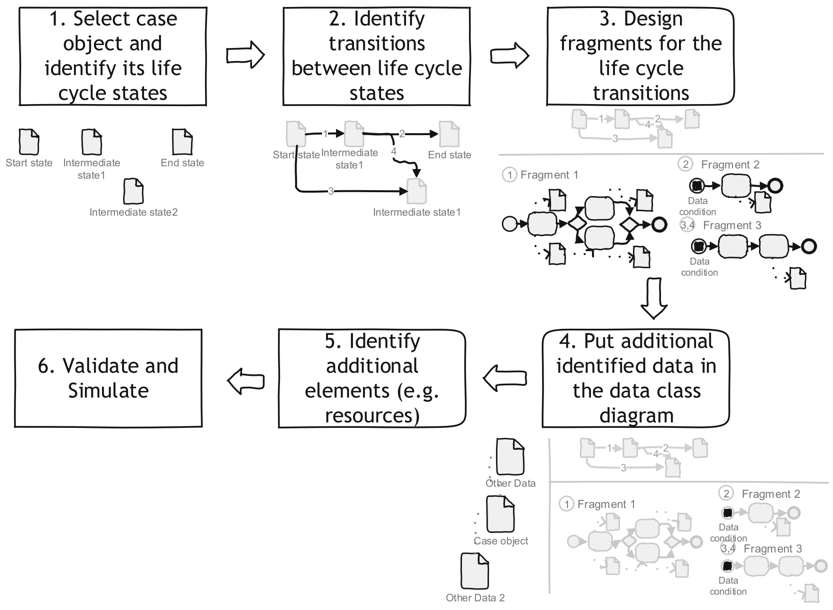


Fig. 4. The OLC of the *Offer* object, case object of the *Offer creation process*, with a fragment realizing transitions 3 and 4

state transitions of the case OLC can be then detailed by fragments. Figure 4 shows, for instance, a fragment for cancelling the offer realizing the state transitions 3 and 4 of the given OLC. This fragment has no start condition, because the cancellation is possible in any state until the goal state is reached. The final states of the OLC define the goal state of the case model; in this case, the goal state results in `Offer[sent] OR Offer[cancelled]`. The steps of the *OLC-first* method are visualized in Fig. 5.

In the first step of this method, the case object – the object living throughout the whole case – has to be identified. Similar to the *process-first* method, the process boundaries are defined by identifying the start and end states of the case



1. Select case object and identify its life cycle states
2. Identify transitions between life cycle states
3. Design fragments for the life cycle transitions
4. Put additional identified data in the data class diagram
5. Identify additional elements (e.g. resources)
6. Validate and Simulate
Start state
Intermediate state1
End state
Intermediate state2
Fragment 1
Fragment 2
Fragment 3
Data condition
Other Data
Case object
Other Data 2

Fig. 5. Steps of the *Object lifecycle-first* case modeling method

model. The end states, thereby, represent the *goal state* of the case model. Next, important intermediate states of the case object during case execution shall be defined by the stakeholders. In the second step, the allowed transitions between the states are defined. The transitions should be either labeled or numbered for assigning fragments to them afterwards. In the third step, a process fragment has to be designed for each data state transition defined in the OLC. Thereby, the start condition of the fragment should be the data input state of the transition and it should return as output the data object in the defined output state of the respective transition. However, one fragment can also be used to realize several transitions as shown in the example of Fig. 4. Then, the start condition of the fragment has to be relaxed for covering multiple transitions (i.e., no start condition, if the fragment can be started in each data object state, or a disjunction of several data object states). If during the modeling other important data objects are found to be of relevance for the fragments, those can be added to the domain model having the case object in its center. In the fifth step, additional elements, for instance, resources, can be added to the case model. Finally, a validation and simulation of the case model should be conducted by the stakeholders for checking its completeness.

The result of this method is a case model on an operational level which can be used for documentation purposes, for redesign efforts etc. In case of an implementation of the case model, e.g. in the Chimera case engine [17,18], it needs to be detailed by providing OLCs for all data objects. Further, the fragments and the domain model need to be extended and adapted with technical aspects needed for the implementation. In comparison to the *process-first* method, the resulting case model is more complete, consisting of a OLC for the case object, a domain model with the case object in the center being related to other needed data, a set of fragments realizing the allowed actions on the case object, and the goal state of the case model.

5 Evaluation

The design and investigation of an artifact in a specific context defines design science [8]. For evaluating our artifacts, the proposed elicitation methods for case models, they were applied in the context of knowledge workers discovering a case model. Case management considers the variabilities of knowledge workers in executing a case. To capture these differences and agreeing on a case model efficiently, we decided to hold one workshop for each method. Also, conducting workshop was preferred as the proposed methods are highly inspired from the workshop-based t.BPM method. This section introduces the experiment design in Sect. 5.1, followed by the two workshop results in Sects. 5.2 and 5.3, and a discussion of them in Sect. 5.4.

5.1 Experiment Design

For each workshop, three participants and one moderator were present. All participants are researchers in the field of business process management with a similar level of modeling expertise. All the participants were unaware of the methods prior to the workshop. At the beginning of both workshops, the method was demonstrated with a small example, and then the actual modeling work began. To analyze the methods later, the workshops were photographed and recorded with the consent of the attendees.

In both workshops, the participants had the task of modeling the process of *organizing a business trip* following the respective method. The process is elicited from the viewpoint of a PhD student or an employee. Essentially, the organization should include getting the consent of the supervisor, planning and booking the trip as well as filling out the travel application (`Dienstreiseantrag` in German). The travel plan can be updated several times and can be cancelled if needed. The process was well known to the participants since they have performed the activities several times for themselves. The moderator helped to follow the method steps and made sure to bring the attendees to an agreement in case of having different opinions about part(s) of the process. Following the t.BPM method, the elicited process fragments were simulated after modeling to assure correctness of the case model.

To evaluate the methods in a consistent way, a list of parameters consisting of qualitative and quantitative measurements were prepared. From the moderator's observation during the workshop and afterwards, from the video recordings, the following five questions were answered:

- Were the participants able to follow the method correctly?
- How was the quality of discussion during workshop?
- How easy was it to identify the flexible process parts?
- How much did the moderator have to intervene?
- Did the participants agree with the resulting case model?

On the other hand, the time recorded for each workshop and the resulting case model answered the following quantitative measures:

- Time to explain the method
- Time to create the complete case model
- No. of fragments and activities per fragment
- No. of data classes and corresponding states
- Object lifecycle completeness
- Model completeness

These will be discussed in detail in the remainder of this section.

5.2 Results: *Process-First* Method

In the first workshop in which the *process-first* method was applied, the process was scoped to start with the event *intent to travel* and end with the event *travel started*. Originally, the *process-first* method postulated that flexible parts have to consist of at least two activities. But while marking flexible activities in Step 2 (see Sect. 4.1), it was detected that flexible parts can contain a single activity only. Following the design science process, we improved the method by allowing the extraction of single-activity-fragments.

The identification of flexible process parts was challenging for the participants at the beginning, but it became easier once single-activity-fragments were allowed. Other than that, there was no difficulty following the method. During the workshop, the participants had lively and detailed discussion about the process scope, activities, events and data objects associated with the case model. The moderator had to intervene only two times when the focus shifted from modeling the case to discussions about the semantics of certain BPMN elements. At the end of the workshop, the participants agreed on the resulting case model.

The main process fragment, *Business trip organization* in Fig. 7, starts with the intent to travel to which the supervisor needs to agree beforehand. This fragment deals with the key activities like preparing the travel plan, booking the conference, accommodation and transport as well as arranging the visa, and notifying the group about the planned absence. Submitting the travel application and canceling or updating the trip were identified as the flexible behaviors. Thus, the fragments *Official confirmation*, *Trip cancellation* and *Trip modification* can be performed at different points of time during the process execution, if certain data conditions are fulfilled. For example, the submission of the travel application can only be done, if the travel plan is in state *created*. There is an option of requesting advanced payment of 75% which can be done at any point as soon as the `Dienstreiseantrag` is submitted. Often business trips are connected with additional vacation days at the conference location such that a fragment for a leave application was designed as well. As a result, five additional fragments with one or more activities were designed.

Based on the case context, it can be observed that the main data class is `Travel plan`. The OLC for this case object was extracted from the given process fragments afterwards, as shown in Fig. 9a. It is not complete as the *update* state is not connected with any other state. Three other data classes, namely, `Dienstreiseantrag`, `Pre-payment` and `Confirmation` were identified additionally during Step 4 of the prescribed method. The quantitative results of the workshop are listed in Table 1 and are further discussed later in this section.

5.3 Results: *OLC-First* Method

The second workshop was held with the same setup and with a disjoint group of participants. To aid comparability of the resulting models, the moderator dictated the main class, as well as initial and ending states. It was observed that focusing on a single data class results in keeping the scope narrow and the level of detail appropriate. On the contrary, the lack of a “catch-all transitions” notation was noticed, since cancellation of the process was possible

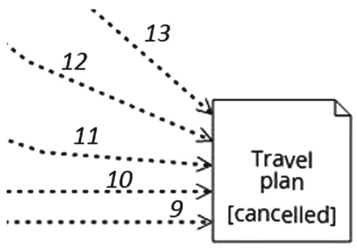


Fig. 6. Catch-all transitions

at any time, resulting in multiple arrows to a single state, as displayed in Fig. 6. Therefore, the initial idea of having an 1-to-1 mapping between transitions and fragments was adapted to an n-to-1 mapping. This dramatically decreased the number of fragments from 12 (1-to-1 mapping) to 5 (n-to-1 mapping).

Given the familiarity with the process and modeling expertise of the participants, it was not difficult for them to follow the method. The steps were executed with enthusiastic discussions and the moderator did not need to intervene at all. There was ample discussion before defining each state, and before drawing each transition. However, once the transitions were there, identifying fragments was easy and fast. The participants were satisfied with the resulting model.

The lifecycle for the case object *Travel plan* is presented in Fig. 9b. The associated fragments are shown in Fig. 8. The fragment *Business trip organization* contains an ad-hoc sub-process [2] whose activities can be executed in random order, multiple times or not at all. In this fragment, the trip is prepared and approval of the travel plan by the supervisor is sought, which can be granted or refused. This fragment covers the state transitions *1–6* and can be repeated, whenever the travel plan is in any of the states *created, confirmed, rejected, booked*. The fragment *Trip booking, modification & Advance payment* take the confirmed travel plan and perform different booking options, again using an ad-hoc sub-process. Transition *7* is mapped to this fragment. The flexible cancellation of the trip is captured with fragment *Trip cancellation* covering the catch-all transitions *9–12*. Submitting the `Dienstreiseantrag` does not change the state of *Travel plan*, as shown in fragment *Official confirmation*. The trip is initiated when travel starts, depicted by fragment *Trip initiation*. This takes the booked travel plan and completes it (transition *8*). As before, the quantitative observations can be found in Table 1.

Table 1. Observations from the two workshops

Observation	Process-first	OLC-first
Time to explain	15 min	20 min
Time to model	61 min	43 min
No. of fragments (activities per fragment)	6 (7,3,1,1,1,1)	5 (4,3,1,1,1)
No. of data class (total states)	Travel Plan (4)	Travel Plan (7)
	Dienstreiseantrag (2)	Dienstreiseantrag (1)
	Prepayment (1)	
	Confirmation (1)	
OLC completeness	Incomplete	Complete
Model completeness	Complete (detailed)	Complete

5.4 Comparison and Discussion

The quantitative metrics presented in Table 1 show that the *OLC-first* method was slightly faster than the *process-first* method. Though the number of

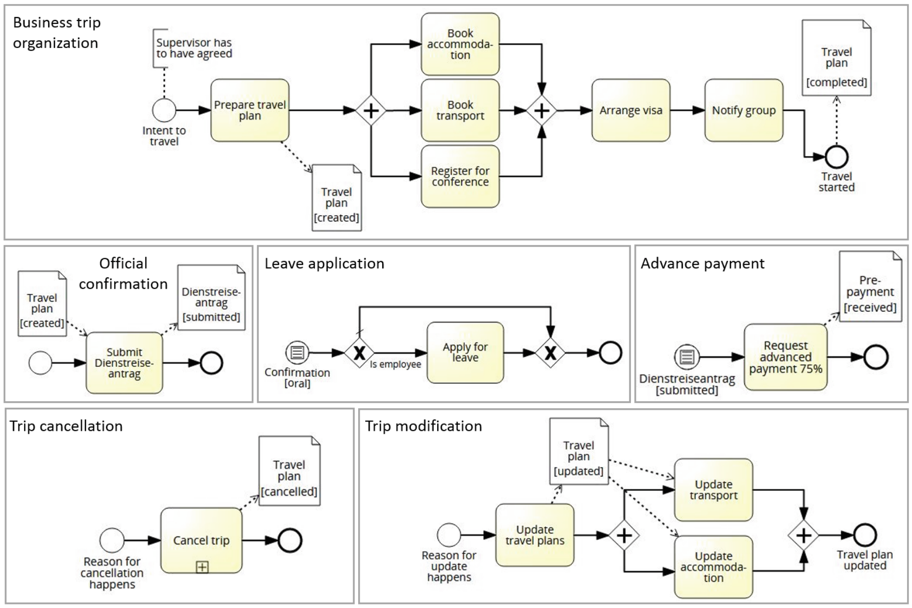


Fig. 7. Resulting case model of the 1st workshop using *process-first* method

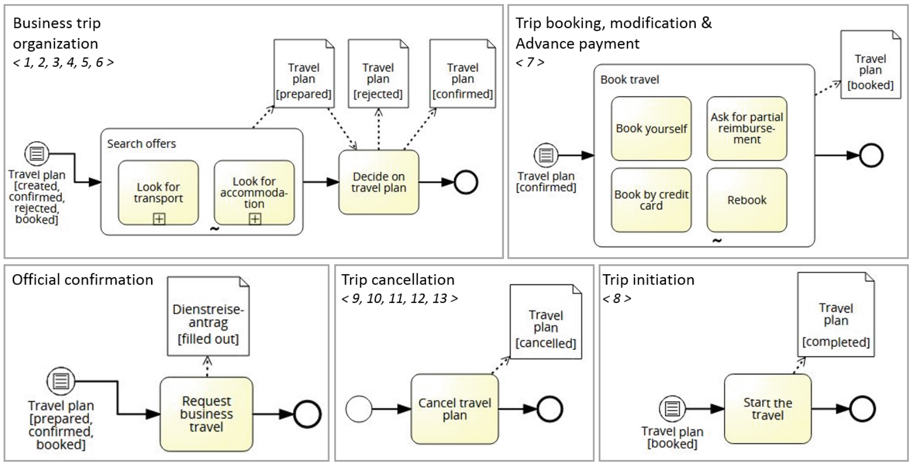
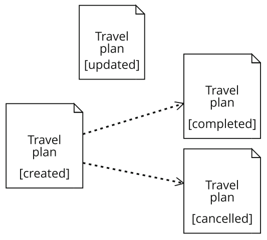
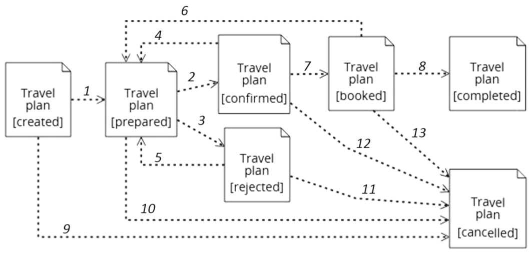


Fig. 8. Resulting case model of the 2nd workshop using *OLC-first* method



(a) Process-first method



(b) OLC-first method

Fig. 9. Data class *Travel plan* with state transitions

fragments is almost same for both methods, the number of activities per fragment shows a significant difference. The *process-first* method resulted in a comparatively larger fragment representing the key activities and some additional smaller fragments representing the variations (Fig. 7). This is helpful to have an overview of the main process flow along with the flexible parts. Whereas, the second method resulted in smaller fragments covering the state transitions (Fig. 8). This, on the other hand, enables reuse of fragments and thus enhances flexibility more efficiently. For example, unlike Fig. 7, the booking activities from the main fragment as well as fragments *Trip modification* and *Advance payment* are combined in one fragment in Fig. 8. In total, *process-first* and *OLC-first* methods cover the whole process with 14 and 10 activities, respectively.

The first workshop had four data classes (Fig. 9a) whereas the second one had only two (Fig. 9b). The possible reason can be the dedicated focus on the main data class which also leads to a more detailed state transitions for the data class `Travel Plan` in the second workshop, resulting in seven states instead of four. Focusing on the object lifecycle makes the second method less vulnerable to errors such as missing transitions, having unreachable states or inconsistent OLC. For the first method, this needs to be checked separately.

As discussed in Sect. 3, data classes together with their associated object lifecycle play an important role in case modeling. The data state conditions define the case flow, enable interactions between the fragments, and also set the case termination criteria. Being a data-centric method, *OLC-first* method assures OLC completeness of the case object. Unlike the other method, it allows greater reuse of fragments and makes the termination criteria explicit. Considering the results and above discussion, we recommend this as a preferred method for case elicitation. However, many organization will have existing process models to which variations should be introduced. Sometimes, it is also hard for the domain experts to estimate how much flexibility is needed, i.e. it is not clear if the process contains enough flexible behavior to model it as a case. In these situations, using the *process-first* method will be more helpful.

While the workshops highlighted many interesting observations and helped to evaluate the proposed methods, they also brought up a few aspects to analyze in future workshops. With respect to scoping the case, in the first workshop, the participants came up with the process boundaries, i.e. the start and end events. Whereas, in the second workshop they were given by the moderator. Again, in the first workshop, leave application was included in the case whereas the participants in the second workshop did not think about it. As the results suggest, the focus on one main data class might prevent the participants to identify other data classes associated with the case. The use of ad-hoc processes in the second workshop added to the convenience of defining fragments. However, since ad-hoc processes are not part of the Chimera approach, they need to be split into further fragments while implementing the model. Last but not the least, the methods have been tested in only one workshop, with a small sample size of three participants. Further evaluations are needed to strengthen the results and if needed, to refine the methodology further.

6 Conclusion and Future Work

In this paper, two methods were presented for case model elicitation: the *process-first* method and the *object lifecycle-first* method. Whereas the first method focuses on a main process from which flexible behavior can be outsourced to smaller process fragments, the second method is driven by the data of a case. It centers around the lifecycle of the case object and designing the fragments that realize its life cycle state transitions. Both methods were evaluated by applying each in a t.BPM workshop eliciting the same business scenario.

The participants of both workshops were satisfied by the created case model. The results of the evaluation show that the process-first method is useful for situations where a main process exists with some flexible behavior. As this method is mainly focused on activities, inconsistencies within the data object lifecycles can occur, e.g. an incomplete lifecycle. This needs, in a second step, a validation and correction of the resulted case model. In comparison, the *object lifecycle-first* method seems to provide a more flexible and consistent case model where fragments can be reused and combined in different ways. In future, we plan to conduct more workshops with other groups of participants to validate the mentioned results. Currently, the methods handled the resource view as second class modeling construct. We want to investigate in future research the role of the resource view in case models.

Both methods were developed for the fragment-based case management approach Chimera. However, the proposed methodology could be also re-used for other fragment-based approaches. For instance, the second method could be adapted for PHILharmonic Flows as follows. As this approach has a strong focus on the domain model, we suggest to start the elicitation with it, instead of the case object lifecycle. In the following steps, fragments should be designed for the data classes' micro-processes, as well as for the macro-processes capturing the inter-relations. This needs further evaluation in future work.

References

1. Marin, M.A., Hauder, M., Matthes, F.: Case management: an evaluation of existing approaches for knowledge-intensive processes. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 5–16. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_1
2. OMG: Business Process Model and Notation (BPMN), Version 2.0, January 2011. http://www.omg.org/spec/BPMN/2.0/
3. OMG: Case Management Model and Notation (CMMN) 1.0. standard formal/2014-05-05, Object Management Group, May 2014
4. van der Aalst, W.M.P., Weske, M., Grünbauer, D.: Case handling: a new paradigm for business process support. Data Knowl. Eng. **53**(2), 129–162 (2005)
5. Hewelt, M., Weske, M.: A hybrid approach for flexible case modeling and execution. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNBIP, vol. 260, pp. 38–54. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45468-9_3
6. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management, vol. 1. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
7. Lübbe, A.: Tangible Business Process Modeling. Dissertation, Universität Potsdam, November 2011
8. Wieringa, R.J.: Design Science Methodology for Information Systems and Software Engineering. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43839-8
9. Peffers, K., Tuunanen, T., Rothenberger, M.A., Chatterjee, S.: A design science research methodology for information systems research. J. Manage. Inf. Syst. **24**(3), 45–77 (2007)
10. van der Aalst, W.M.P., Barthelmess, P., Ellis, C., Wainer, J.: Proclets: a framework for lightweight interacting workflow processes. Int. J. Coop. Inf. Syst. **10**(04), 443–481 (2001)
11. Künzle, V., Reichert, M.: PHILharmonicFlows: towards a framework for object-aware process management. J. Softw. Maint. Evol. Res. Pract. **23**(4), 205–244 (2011)
12. Nigam, A., Caswell, N.S.: Business artifacts: an approach to operational specification. IBM Syst. J. **42**(3), 428–445 (2003)
13. Bhattacharya, K., Hull, R., Su, J.: A data-centric design methodology for business processes. In: Handbook of Research on Business Process Modeling, pp. 503–531 (2009). Chapter 23
14. Combi, C., Oliboni, B., Zardini, A., Zerbato, F.: A methodological framework for the integrated design of decision-intensive care pathways–an application to the management of copd patients. J. Healthc. Inform. Res. **1**(2), 157–217 (2017)
15. van der Aalst, W.M.P.: Process Mining: Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
16. Weske, M.: Business Process Management - Concepts, Languages, Architectures, 2nd edn. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28616-2
17. Haarmann, S., Podlesny, N., Hewelt, M., Meyer, A., Weske, M.: Production case management: a prototypical process engine to execute flexible business processes. In: Proceedings of the BPM Demo Session, (BPM 2015). CEUR Workshop Proceedings, vol. 1418, pp. 110–114. CEUR-WS.org (2015)
18. Beyer, J., Kuhn, P., Hewelt, M., Mandal, S., Weske, M.: Unicorn meets chimera: integrating external events into case management. In: Proceedings of the BPM Demo Track. CEUR Workshop Proceedings, vol. 1789, pp. 67–72. CEUR-WS.org (2016)

An Experience Report on Modeling Collaborative Processes (BPMDS 2018)

Modeling Collaborative Processes with CMMN: Success or Failure? An Experience Report

Ioannis Routis(✉), Mara Nikolaidou, and Dimosthenis Anagnostopoulos

Harokopio University of Athens, 70, Eleftheriou Venizelou Av., 17671 Kallithea, Greece
{i.routis,mara,dimosthe}@hua.gr

Abstract. Case Management Modeling and Notation (CMMN) standard was introduced as a alternative language of Business Process Management Notation (BPMN) targeting the modeling of human-centric processes characterized by agility. During our involvement in a project for modeling a collaborative process, the way CMMN may be used by modelers was evaluated. The report presents the experience gained from such an attempt, comparing two different models designed by different modeling groups and discuss their design perspective. Answering whether using CMMN by each of the modeling groups to model collaborative processes led to a success or to a failure is not straightforward. The report identifies the resulting issues of each of the perspective, providing some ideas on how these issues may be addressed.

Keywords: Process modeling · Collaborative processes
Case management modeling and notation · Executable models
Experience report

1 Introduction

Collaborative processes are recursive ones where two or more people or businesses work together toward an intersection of common goals by sharing knowledge, learning, and building consensus, according to Wikipedia. Business Process Management Notation (BPMN), well-known for its structural philosophy, is not efficient for modeling such processes, as BPMN work-flows are designed to be strict, unadaptive to changes as well as not supportive to decision-making and collaboration [12]. On the other hand, alternative-to-BPMN languages such as Case Management Modeling and Notation (CMMN) tend to facilitate and support organizations that would prefer to view their process as cases, where participants exchange data and ideas to fulfill a specific goal, e.g to "close" the case, while there are no strict rules restricting their interaction [8]. One of the main differences between CMMN and languages like BPMN is the paradigm shift from prescriptive to declarative [2]. This differentiation in modeling philosophy was the major issue we faced when requested to participate in a project to consult a public organization on exploring a collaborative process and develop the appropriate software to support it.

More specifically, we were involved in ReWeee Initiative employed by Appliances Recycling SA, alongside with more the twenty of its collaborators both Greek and foreign, to promote electronic product exchange and consequently reduce electronic waste. We were assigned the design and implementation of a collaborative platform to promote electronic equipment exchange. Our first task constituted of the user requirements analysis stage, where we should identify how the electronic equipment exchange should take place and the way a software platform may promote the overall process. All the partners involved were participating in the requirement analysis process and apparently argued on the way exchange should be performed. Thus, we identified the need to provide them with a model of the process to promote discussion. For the purpose, we decided to employ Case Management Modeling and Notation as an alteration to BPMN, an attempt also made in [10].

According to its standard, CMMN, was primarily designed supplementary to BPMN, in order to model and analyze parts of a process, better suited to be handled as Cases. Such an attempt of combining BPMN with CMMN in order to model highly flexible and variable processes was done in [13]. In our case, the goal was to utilize CMMN as a standalone methodology for modeling a collaborative process, like electronic equipment exchange. However, with being a new standard, there currently was some uncertainty about the applicability of CMMN in real-world scenarios as is also claimed in [5].

The rest of the paper is structured as follows. In Sect. 2, details about the ReWeee Project and the collaborative platform that we had to develop are provided, presenting the requirements that should be satisfied referring to both user-to-user interaction and user-platform interaction. In Sect. 3, the modeling notation for Case Management is briefly presented, namely, its core features and how these can be combined in order to model a process. In Sect. 4, we discuss the modeling process using CMMN. Different modeling teams designed alternative models for the exchange process. Their different perspectives were identified and the design philosophy of each model is annotated. In Sect. 5, the questions set to the modeling teams are presented, along with the answers given, while further discussion takes place identifying issues that need further exploration towards CMMN adoption. The final section refers to the conclusions that can be drawn from this experience report, setting future work for further research and exploration by the authors.

2 ReWeee Initiative

The LIFE ReWeee Initiative aims to prevent the creation of Waste Electrical and Electronic Equipment (WEEE) and is co-ordinated by employed by Appliances Recycling SA. This initiative includes a major action promoting the donation and exchange of EEE in a national-scale fashion, while a web-based collaborative platform, namely the ReWeee platform, should be developed to bring together organizations and individuals participate in this action [1]. The main goal of the platform to facilitate and promote Electrical and Electronic Equipment exchange and donation among households or households and public/private bodies. Its success lies within the social communication between volunteers and their collaboration in order to achieve the best possible result.

2.1 Case Description: EEE Exchange

We considered the EEE exchange process as a case to be completed with the collaboration of all interested parties. Our first task was to obtain an abstract description of the case from Appliances Recycling SA. When talking about a collaborative platform, our client had in mind a short of social application or wiki, thus, this is the first-level description of EEE exchange case as provided to us. When described by ReWeee management team, a user-centered approach was followed [11], based on the assumption that their were prescribing what different kinds of participants should do in the context of a collaboration environment.

"First of all, there are two types of users. These are guest and registered users that differentiate themselves in the permissions that they get granted as far as the use of the platform is concerned.

More specifically, when any unregistered user visits the web platform for the first time, he gets prompted to register, by creating a user account. This account can be created either by signing up via an email and a password or by signing up via a social network account, which requires his/her giving to the platform the necessary permissions for using personal data. After a successful registration, the, from now on, registered platform user, is able to submit an advertisement donating or exchanging an item, to declare interest for an existing EE product and propose an offer to acquire it, as well as to communicate with any other user who owns a desirable electric device.

Moreover, a registered user is not only able to search a product based on some conditions, namely, filters like item categories, item state, donating-user region, but also to either suggest changes regarding the item's category for which he/she is searching, or even to comment in an advertisement that he/she had expressed interest for. That way, the appropriate users will be notified for either the category change proposal or the commenting in an advertisement.

Finally, registered users have a profile in which they are able to be notified for any recycling actions taken via a news-feed as well as being informed for general topics regarding recycling and its benefits. Within each user's profile, a calendar exists via which a user can be informed for any recycling events taking place."

3 CMMN Language

The focus of the CMMN specification is the notation, the meta-model, interoperability between tools, and minimum execution semantics [7]. The main objective of Case Management Modeling and Notation is to define a common meta-model and notation for modeling and graphically expressing a Case. A Case involves actions taken regarding a subject in a particular situation to achieve a desired outcome. Traditional examples are Cases that refer to legal and medical working environments, where a legal Case involves the application of the law to a subject in a certain fact situation, and a medical Case involves the care of a patient in the context of a medical history and current medical problems. The subject of a Case may be a person, a legal action, a business transaction, or some other focal point around which actions are taken to achieve an objective. The situation commonly includes data that inform and drive the actions taken in a Case [9].

A case contains all elements of a CMMN model. There are two phases for each case. Design-time phase, during which, business analysts prepare the case execution by modeling the case. Once a case has started to being executed, the case is in the run-time phase. In this phase, the case workers are working on achieving the case objectives [5]. A CMMN model primarily comprises the following case items:

Task: Tasks describe activities that can be executed during the run-time phase. Four types of tasks are supported: human (performed by a knowledge worker), process (to embed a process, e.g. a BPMN model), decision (to embed a decision, e.g. a DMN model) and case (to embed other cases e.g. other CMMN models);

Stages: Stages are containers for case elements. They allow structuring a case hierarchically.

Milestones: Milestones represent a target and thereby allow checking the progress of a case.

Event Listener: An Event Listener captures events, which are things that "happen" during a case. Events may trigger, for example, the enabling, activation, and termination of stages and tasks, or the achievement of milestones.

Sentries: Sentries allow defining temporal-logical dependencies between stages and/or tasks, "watching out" for important situations to occur (or events), which influence the further proceedings of a case. If a stage or task has a sentry attached, it can only be started if the precondition defined by the sentry is fulfilled. Sentries also represent a combination of conditions and events that define the sequence of tasks to be implemented [11].

Case File Item: A Case File Item represents a piece of information of any nature, ranging from unstructured to structured, and from simple to complex, and can be defined based on any information modeling language. In knowledge-intensive work, documents are typical outputs of tasks or stages.

Connectors: Connector serves to visualize complex dependencies between elements; It also represents the standard events that drive the flow of the Case.

Discretionary Items: These identify an item, of which instances can be planned, to the "discretion" of a case manager [2,5].

Case management is concerned with determination of which tasks are applicable, or which follow-up (discretionary) tasks are required, given the state of the Case. Decisions and flow may be controlled by events or new facts that continuously emerge during the course of the Case, such as the receipt of new documents, completion of certain casks, or achieving certain Milestones. Individual tasks that are planned and executed in the context of the Case might be predefined procedural processes in themselves, but the overall Case cannot be orchestrated by a predefined sequence of tasks. Finally, the meta-model and notation are used to express a case model in a common notation for a particular type of Case, and the resulting model can subsequently be instantiated for the handling of a particular instance of a Case [9].

4 Modeling EEE Exchange with CMMN

Based on the description provided in Sect. 2.1, one could easily conclude that EEE exchange could not be effectively modeled using BPMN, as the sequence of activities unpredictable and random. However, a visual model of the process would be useful during user requirement analysis stage. For that reason, we identified it as a Case that could be modeled using CMMN, which enables the modeling of such activities in a more fluid fashion.

To evaluate whether CMMN is easy to use by modelers, two modeling groups we established, consisting of a BP modeling expert, a junior modeler and a developer each, all familiar with BPMN. None of them was familiar with CMMN. Both groups were provided with a short case description (see Sect. 2.1) and a chance to interview ReWeee management team. The purpose of the study was to identified potential obstacles in CMMN adoption. The groups studied CMMN standard and relevant literature and resulted in a EEE exchange case model using CMMN. Both of them interviewed ReWeee experts and ask for additional information on the provided description. No interaction between the modeling groups was allowed. The two models produced were extensively different, a not anticipated outcome, as discussed in the following. We consider both models to be valid in terms of utilizing CMMN concepts. Though, they project two different perspectives as far as both the level of analysis and design philosophy is concerned, as shown in Figs. 1 and 2.

4.1 Analytical Perspective

The first one of the models is described in Fig. 1 below. It presents the functionality that the collaborative platform provides to its users, highlighting the interaction between the various activities included in the ReWeee Case.

What can be commented for this modeling attempt, is that its design logic was based in analytically depicting what the user of the collaborative platform should be enabled to do, as narrated by ReWeee management team. As the end-users had in mind a description of the case as a sequence of available screens by a collaboration platform, the modeling team adapted their perspective. What is primarily modeled is which are the events that lead the case from one state to another (in practice from one screen to another) without having in mind which activities are mandatory or not. For that reason, the is no use of discretionary tasks [9], while the majority of actions are linked to each other. Finally, data

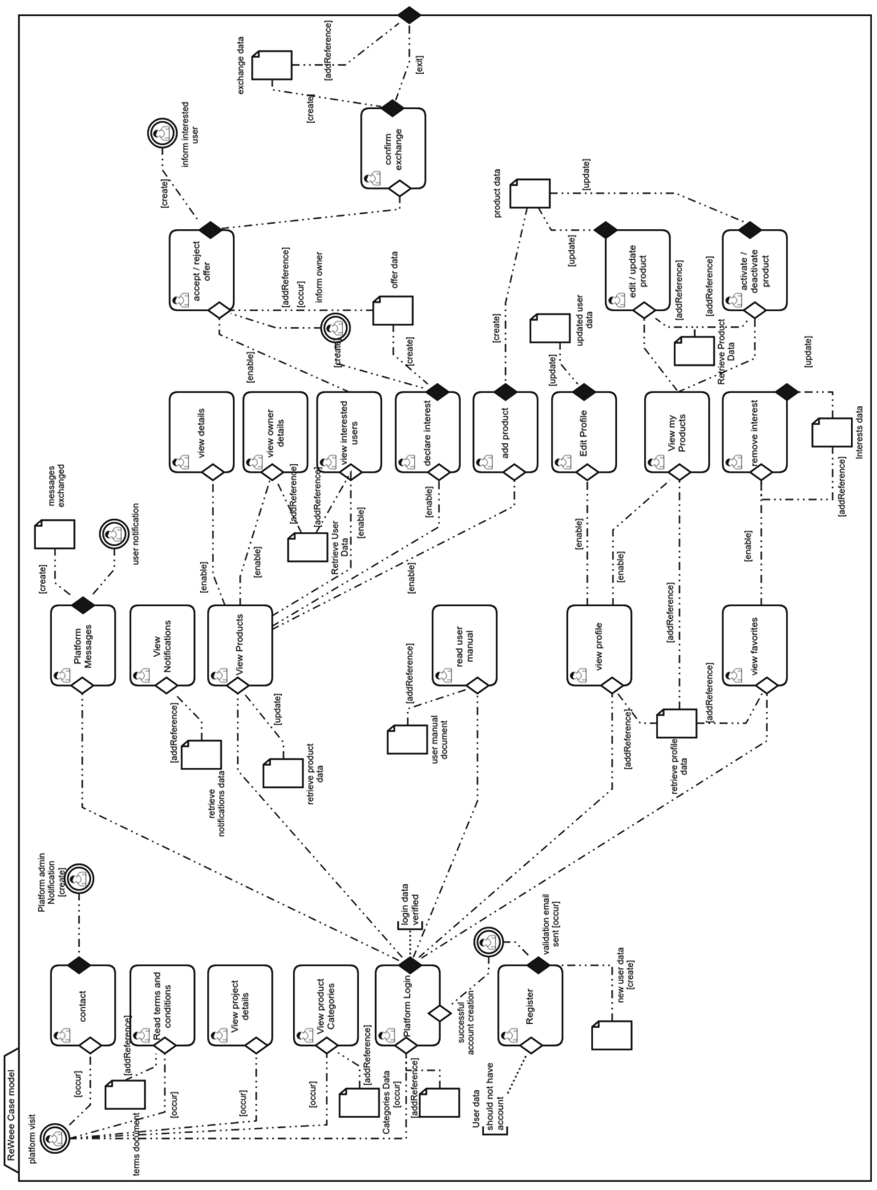


Fig. 1. EEE exchange case from an analytical perspective.

queried during the platform life cycle are emphasized, while EEE exchange case milestones are not modeled. It is attempted to describe the case in detail so as to ensure that nothing is going to be skipped during implementation.

4.2 Abstractive Perspective

The second model for the EEE exchange Case, is shown below on Fig. 2. It also projects, like the first one, the whole functionality provided from the ReWeee platform, but in contrast with the first one it emphasizes on which actions are mandatory for the EEE exchange case to be completed successfully. More analytically, there is an extensive use of discretionary tasks, while there is a large amount of activities that are not connected with each other. Data created during the Case lifecycle are hardly under consideration in this model, but on the contrary milestones are defined to highlight anything that is considered as important for the ReWeee Case, while stages are also modeled in order to isolate the less important parts. What could be commented is that this modeling attempt comprehends CMMN nature better, but the model created it is quite likely to deviate from the actual implementation as it is not taken under consideration. Contrariwise, it is attempted to ensure that the end-user has understood the EEE exchange case components.

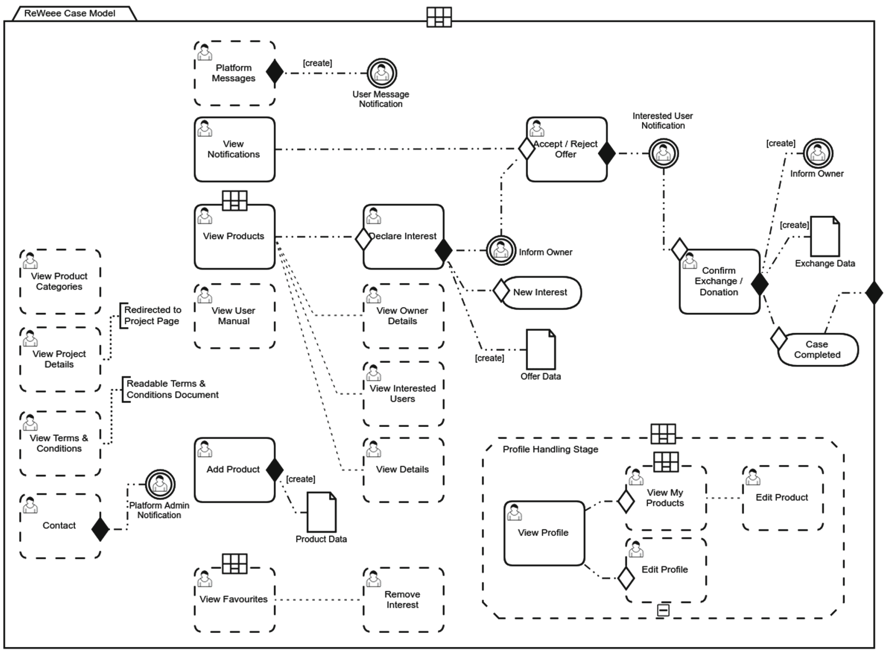


Fig. 2. EEE exchange case from an abstractive perspective.

5 Reflections

5.1 Rising Questions

As the first model (see Fig. 1) seems to be more complex and structural that the second one (see Fig. 2), one could argue that the first modeling team fail to understand how to use CMMN, while the second one succeeded. In the following, we discuss the results of interviewing both teams and explore what drove them to adopt their perspective. At the end of our discussion, we concluded it is not as straightforward as it seemed to explicitly choose or dismiss one perspective. The questions set to two modeling groups are presented, alongside with the arguments given from them. The modeling experience of both groups was discussed, with some queries as a basis, in order to explore their willingness to adopt CMMN in modeling similar Cases. These queries referred to: the modeling objective of their attempt, what difficulties they faced in modeling the Case and what would be the difference in modeling the EEE exchange Case with BPMN. The following table (Table 1) presents a summary of the arguments provided from the modeling groups.

Table 1. Queries and Arguments for the two modeling perspectives

Queries	Analytical perspective	Abstractive perspective
Modeling target	Design the EEE exchange case, as described by end-users, representing tasks in detail so as to ensure the optimal implementation	Define the mandatory tasks of the Case to highlight those that when instantiated could lead to Case completion
Ease of modeling	Modeling was mainly facilitated by the use of Sentries. It helped modelers achieve the objective of modeling the sequence of activities	Modeling was mainly facilitated by the use of Discretionary Tasks. Easier highlighting of the mandatory Tasks for the Case completion
Modeling difficulties	Difficulties in understanding how to represent the flow of the Case and to comply with the philosophy of CMMN standard	Difficulties in defining the correlation between the Tasks as well as in identifying how each task can be instantiated
BPMN differentiation	Design philosophy was very close to the one of BPMN as the sequence of Tasks was the main objective of modeling	Platform functionality and Tasks were modeled according to the CMMN standard in a quite different way from BPMN

In principal, the analytical model (Fig. 1) had as its primary aim to describe exchange case flow in detail, having the visitor of ReWeee platform in mind. It was an effort to precisely describe the user experience expected as exposed by ReWeee management team. Emphasis was given to the correlation of tasks representing each possible step of the overall process in great detail. Furthermore, focus is given to depict the way EEE exchange Case will be automated, without missing any of the projected functionality. That leads to a quite descriptive model which seems more relevant to a BPMN model than one modeled with CMMN. On the other hand, the abstractive model (Fig. 2), attempts to define which ones of the Case activities are mandatory so as to complete the Case by making extensive use of the Discretionary Task notation element. That way, a high level of abstraction is provided to the model, making it easily readable and to comply with CMMN philosophy. Though it lacks in information about Case Data or how Model elements are correlated.

Moreover, the ease of modeling was also discussed with the two modeling groups. For the one that modeled EEE exchange Case in an analytical way, Sentries [9] were the most facilitating feature of CMMN, enabling them to model transitions in the case's state. Having the projection of the sequence of activities as their main goal, this feature was the one that helped them the most at their modeling attempt. The other group, that created the abstractive model, find the ease of modeling lying to the use of Discretionary Tasks for modeling less important Tasks. That way, the mandatory Tasks, that lead to Case completion, could be highlighted.

On the other hand, both of the groups referred facing difficulties during the modeling of the EEE exchange Case. For the first group, that modeled the process analytically, it was difficult to identify how to model the flow of the process. Namely, they could not identify easily how sequence is represented. For that reason they utilized extensively the notion of Sentry, using events not only to project the control of flow but also to describe how the completion of a Task could commence another Task. However, that made their model quite complex, not familiar with CMMN. On the contrary, for the other group that modeled the ReWeee process in a more abstractive fashion, it was difficult to define the exact correlation between the Case activities. That led them to model a mass of Tasks, independent to each other, which was quite unusual for the modelers and that also led to an undefined sequence of activities. As it was commented in the model description above, that could likely make the model deviate from the actual implementation.

Finally, concerning the differentiation from BPMN, the analytical model had a design philosophy very close to the one of the BPM, as it had the representation of the sequence of the Case activities as its main objective. Extensive data modeling and strict sequence definition made the modeling very close to the philosophy of BPM. The final model looked different from a corresponding BPMN model just because the differentiation in notation elements. On the contrary, the abstractive model projected the functionality provided by the projection of the Tasks available but in a quite different way from a BPM model. There was not such a thing like sequence representation, while data were completely out of frame. This fact led the model to look like more than a definition of Tasks without taking under consideration the correlation between them. It may followed the philosophy of CMMN but the Case representation was not anywhere near an implementation model.

5.2 Discussion

The two models, seem to be conflicting to each other nearly in every query that was set. Firstly, they had quite opposing perspectives. The first (Fig. 1) seemed to focus on modeling the automation of EEE exchange case, handling tasks in a unified fashion, regardless if these were mandatory or optional for the Case completion and, in general, modeling nearly everything that one would like to know for the Case before automating it. The second (Fig. 2) focus exactly on the opposite. Having the identification of the main tasks and the distinction between mandatory and optional tasks as main objective, it is unimportant for the modeler to represent sequence and correlation of tasks in great detail. The reason can be given through the fact that the second model complies totally with the philosophy of CMMN that, as was mention in its description earlier in this paper, does not focus on defining automation properties. However, one should have in mind the CMMN is claimed to provide for executable models [7].

Different goals, led to different views on the Case modeling. For the first group, the projection of case automation related details was the main reason of finding difficulties in complying with CMMN. Case Management Modeling and Notation addresses Cases from a different point of view from BPMN. While BPMN offers only limited precautions for ad-hoc adaptations, CMMN provides a way for modeling cases where the logical dependencies may be optional [5]. To attempt the substitution of BPMN with an alternative modeling language, is quite tricky since a substitution in modeling philosophy is also necessary. A misunderstanding could lead to have a complex model, not complying with the methodology provided, that has no differentiation with BPMN in the way modeling of tasks is handled. On the other hand, having a model with quite distinct tasks, not connected with each other, made it harder for the modelers to find correlations between the Case activities and highlight how tasks are instantiated. This could not help anyone who would like to automate the case, as important components are missing like data and events.

The discussion above, led as to the conflicting consequence that a descriptive CMMN model like the one of Fig. 1, which could be considered as "executable", looks in way quite alike as a BPMN model, in terms of identification of flow. However in the CMMN model, utilizing Sentries the conditions (e.g. recorded events) to enable a task execution are identified and the events generated by its execution are also recorded. The main difference distinguishing BMPN and CMMN has to do with the way representing flow, in an analogy referring to the difference between UML activity and state diagrams, both representing behavior.

On the contrary, an abstractive model like Fig. 2, leads to a model easily identified as a Case Management model, describing a Case. However, modelers claim automation details might be missing, which are important to implement the ReWeee platform. It should be noted that CMMN adoption should not be restrictive on the selection of the platform used for the case automation.

This contradiction on the way CMMN can be employed is what the authors of this paper identify as the most important issue arising. While CMMN seems ideal for modeling Cases in an declarative manner in design-time, it provides no guidance on how to represent a running case, e.g. a way to ensure that case models could be executable, as is highlighted in [5]. Trying to define such an "executable" CMMN model was in practice the fact that caused confusion in the first modeling group.

Towards this direction, the following questions are identified for future reference.

1. Whether CMMN could depict executable models, in a sense similar to BPMN, that nevertheless will comply with its philosophy should be further explored. Using existing tools, it may only become executable in practice as a supplementary part of BPMN [3], as indicated in its standard.
2. CMMN extensions could be proposed so as to depict executable models, considering the fact that at this stage of research it seems unable to satisfy such requirement. Such an example is examined in [6]. The extension should keep the main philosophy of CMMN unchanged, retaining it declarative, while it should provide automation to promote execution capability. Ideally, such extensions should be general enough to be supported in different case management support platforms.
3. What are the parameters of such an executable CMMN model should be identified, along with its design requirements and its main parameters, namely the main ingredients that could make CMMN models executable. The term "executable" should also be explored in a difference sense complying with CMMN philosophy. While the existence of a CMMN engine may not always be a desired property, the support of task and case templates as well as data management utilities may be of use.

6 Conclusions and Future Work

Our experience using CMMN to model collaborative processes was discussed in this work. What can be concluding is the fact that we have a new modeling notation for flexible and variable process, identified as Cases, which could be accepted as an alternative to BPM methodology at least in modeling the design-time of a process. Considering the difficulties of learning a new non traditional BPM alternative as is mentioned in [4], the CMMN models projected earlier in this work, describe a collaborative process in an acceptable way, without any loss of the activities modeled.

However, according to our experience, it is not obvious how to employ CMMN to model processes or cases, resulting in executable models as provided by BPMN. There have been directions in literature towards this end, and considering its continuously growing reputation Case Management seems promising for providing to process modelers what BPMN lacks of, namely, flexibility in the process modeling. Towards this end, we set as future challenges to find responses to the questions set in the discussion. Additionally, a further evaluation of CMMN language could take place, considering the modeling experience of other modeling groups in similar Cases.

Acknowledgment. This work is co-funded by the European Commission through the LIFE+ Funding program, LIFE14 ENV/GR/000858: LIFE REWEEE.

References

1. ReWeee Project basic information. https://www.reweee.gr/en/basic-information.html. Accessed 22 Feb 2018
2. Carvalho, R., Boubaker, A., Gonzalez-Huerta, J., Mili, H., Ringuette, S., Charif, Y.: On the Analysis of CMMN Expressiveness: Revisiting Workflow Patterns (2016)
3. Juric, M., Bernhardt, S., Normann, H., Schmiedel, D., Schmutz, G., Simpson, M., Winterberg, T.: Design Principles for Process-driven Architectures Using Oracle BPM and SOA Suite 12c. Packt Publishing (2015). https://books.google.gr/books?id=fv0KCgAAQBAJ
4. Koutsopoulos, G., Bider, I.: Teaching and learning state-oriented business process modeling. experience report. In: Reinhartz-Berger, I., Gulden, J., Nurcan, S., Guédria, W., Bera, P. (eds.) BPMDS/EMMSAD -2017. LNBIP, vol. 287, pp. 171–185. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59466-8_11
5. Kurz, M., Schmidt, W., Fleischmann, A., Lederer, M.: Leveraging CMMN for ACM: examining the applicability of a new OMG standard for adaptive case management. In: Proceedings of the 7th International Conference on Subject-Oriented Business Process Management, p. 4. ACM (2015)
6. Marin, M.A., Brown, J.A.: Implementing a case management modeling and notation (CMMN) system using a content management interoperability services (CMIS) compliant repository. CoRR abs/1504.06778 (2015). http://arxiv.org/abs/1504.06778
7. Marin, M.A., Hauder, M., Matthes, F.: Case management: an evaluation of existing approaches for knowledge-intensive processes. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 5–16. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_1
8. Motahari-Nezhad, H.R., Swenson, K.D.: Adaptive case management: overview and research challenges. In: 2013 IEEE 15th Conference on Business Informatics, pp. 264–269, July 2013
9. Object Management Group: Case Management Model and Notation v1.1 (2016). http://www.omg.org/spec/CMMN/1.1/CMMN
10. Plebani, P., Marrella, A., Mecella, M., Mizmizi, M., Pernici, B.: Multi-party business process resilience by-design: a data-centric perspective. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 110–124. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_8
11. Routis, I., Nikolaidou, M., Anagnostopoulos, D.: Using CMMN to model social processes. In: Teniente, E., Weidlich, M. (eds.) BPM 2017. LNBIP, vol. 308, pp. 335–347. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74030-0_25
12. Routis, I., Stratigaki, C., Nikolaidou, M.: Exploring ACM and S-BPM for modelling human-centric processes: an empirical comparison. In: Proceedings of the 8th International Conference on Subject-oriented Business Process Management, p. 5. ACM (2016)
13. Wiemuth, M., Junger, D., Leitritz, M.A., Neumann, J., Neumuth, T., Burgert, O.: Application fields for the new Object Management Group (OMG) standards case management model and notation (CMMN) and decision management notation (DMN)in the perioperative field. Int. J. Comput. Assist. Radiol. Surg. **12**(8), 1439–1449 (2017). https://doi.org/10.1007/s11548-017-1608-3

EMMSAD 2018

The Power/Generality Trade-Off in Decision and Problem Modeling: Theoretical Background and Multi-level Modeling as a Resolution

Alexander C. Bock(✉)

Research Group Information Systems and Enterprise Modeling, University of Duisburg-Essen, Essen, Germany
alexander.bock@uni-due.de

Abstract. A central conflict in decision and problem solving support is known as the 'Power/Generality' trade-off. The incorporation of a high level of domain-specific concepts and mechanisms in a decision instrument will increase the instrument's power but will do so at the cost of the instrument's generality. This paper has two purposes. First, it brings to attention the power/generality conflict in conceptual decision and problem solving modeling, and it demonstrates the resultant problems in relation to an existing enterprise decision modeling language. Second, the paper proposes the use of a multi-level modeling paradigm as a possible resolution of the conflict, and it proposes concrete re-conceptualizations for an existing modeling language to alleviate the associated problems.

Keywords: Decision making · Problem solving
Power/generality trade-off · Decision modeling · Multi-level modeling
Enterprise modeling language

1 Introduction

A central conflict in decision and problem solving support has come to be recognized as the *'Power/Generality'* trade-off [1, pp. 371–373], [2, pp. 606, 611]. The generality of an instrument is the range of problems to which it can be applied [1, p. 371]. The power of an instrument is its "ability to deliver solutions" [1, p. 372]. The power/generality trade-off, then, consists in the fact that "there is an inverse relationship between the generality of a method and its power" [1, p. 372]. The more domain- or task-specific elements an instrument incorporates, the better is the support for problems in that domain, but the more limited is the scope of applicability (e.g., [2, p. 606]). The power/generality trade-off has been noted by Newell as early as 1969 [1, pp. 371–373], and it is found in many fields of inquiry. For example, the conflict is also known as the 'Usability/Reusability' trade-off in method or programming constructs design [3, pp. 915–916], [4, pp. 117–118], [5, pp. 1–2], the 'Semantics/Reuse' trade-off in formal language design [6, pp. 67–70], the 'Reference Modeling Dilemma' [7, pp. 28–29], the 'Generality/Utility' conflict in design-oriented research [8, pp. 19–20], and even the 'Generality/Power' trade-off in empirical theory building [9, pp. 591–592].

A special domain in which the power/generality conflict manifests itself is *enterprise decision modeling*. In the past decade, the field of conceptual modeling has brought forward a number of modeling languages to describe organizational decision situations (e.g., [10–13]). All these languages require the modeler to define decision or problem types at some level of specificity, giving rise to the power/generality trade-off. If the modeler models decision types at a high level of specificity, then more useful information for special situations can be documented, but a great number of decision types have to be defined. Conversely, if the modeler defines decision types at a low level of specificity, then only few types need to be modeled, but the support is restricted to highly general information.

Rather than offering assistance in dealing with the trade-off, contemporary language architectures exacerbate the difficulties. For example, because the traditional conceptual modeling paradigm admits only *one* classification level to model type constructs (e.g., [14, pp. 6–8], [15, pp. 18–20]), decision types of different generality/specificity must all be modeled on the same level. This hampers interpretability and formal discriminability. Furthermore, the limitation to one abstraction level makes it impossible to express instance-of relationships between decision types of different generality/specificity in one model, or to instantiate more specific decision types over time. This restricts model expressiveness and extensibility when seeking compromises in the power/generality trade-off.

In recent years, the paradigm of conceptual *multi-level modeling* has received increasing attention (e.g., [16–18]). Multi-level modeling expands the type modeling space for users to several classification levels [16, pp. 743, 745–746], [17, pp. 321–322]. Notably, one goal that has driven multi-level modeling was to mitigate the power/generality conflict in the design of domain-specific modeling languages (DSMLs) (see [19, pp. 29–31], [17, pp. 319–322]). Rather than forcing the modeler to select a single level of generality/specificity for a DSML ex ante, multi-level modeling enables to gradually refine language constructs over different levels [17, pp. 319–322]. This makes it easier to find power/generality compromises. Modelers can start from general concepts and users can add more specific concepts where needed. Given this capability, multi-level modeling promises to hold a remedy for the power/generality conflict in decision modeling as well.

This paper has two purposes. First, it seeks to bring to a wider attention the power/generality trade-off in conceptual decision and problem solving modeling, and to demonstrate the entailed problems with the example of an existing enterprise decision modeling language. Second, the paper proposes the use of a multi-level language architecture as a possible remedy for the power/generality trade-off, and it presents concrete re-conceptualizations for the existing modeling language to harness these potentials. Thus the contribution of the paper is twofold. It advances the theoretical discussion about conceptual decision and problem modeling and it further develops an existing modeling language.

The paper proceeds as follows. Section 2 discusses the theoretical background, while Sect. 3 illustrates the practical effects of the power/generality trade-off. Section 4 explains the use of multi-level modeling as a resolution and presents new constructs for a decision modeling language. The paper closes in Sect. 5.

2 Theoretical Background

This section establishes a foundation for the paper. It first theoretically introduces the power/generality conflict (Sect. 2.1) and then offers a brief overview of decision and problem solving research (Sect. 2.2).

2.1 The Power/Generality Trade-Off and Its Variants

The power/generality trade-off is highly universal and has been detected in different areas of study. In essence, the trade-off is an *inverse* relationship between features of artifacts on two different dimensions (cf. [1, p. 372], [4, p. 125]). Following Newell, the first dimension is here called *power* [1, p. 372]. This dimension can be summarized as an artifact's effectiveness, or overall capability to meet the desired purposes. For example, in the realm of computerized problem solving research, where the trade-off was articulated early on, the power of a method can be seen as its "ability to deliver solutions" [1, p. 372] or the "useful support [it] can provide for problems" [2, p. 606]. In other areas, other estimates are used. For example, in the area of programming constructs, the dimension has been called 'usability' and linked to a construct's capability to support task automation [4, p. 118]. In the area of reference modeling, the dimension has been understood as a model's "fit to [...] individual requirements" [7, p. 28]. Importantly, the power dimension has also been applied to modeling languages [19, pp. 29–30], [17, p. 320]. Here it has been called, for example, 'potential productivity gain' and defined as the "contribution to productivity and integrity" [19, p. 29]. Evidently, the 'power' criterion is highly broad. Being a blanket measure for an artifact's effectiveness, it summarizes many individual properties. In most cases, different operationalizations of 'power' will be conceivable (e.g., [1, p. 372]).

Again following Newell, the second dimension is here called *generality* [1, p. 371]. Alternative names include 'reusability' [3, p. 915], [4, p. 118], [5, p. 2] and 'scale of reuse' [19, p. 29]. This dimension is defined rather uniformly across research fields. It describes the "range of problems for which [a method] is applicable" ([2, p. 606]; for similar definitions, see [4, p. 118], [8, pp. 19–20], [19, p. 29]). For example, in the realm of conceptual modeling, the Entity-Relationship Model [20] can be viewed to possess a high degree of generality; it can be used to describe states of affairs in almost any industry. In contrast, a specialized language such as the OMG Decision Model and Notation (DMN) [10] is restricted to a narrower area of interest and cannot be applied elsewhere. This contrast corresponds to the well-known distinction between general-purpose and domain-specific modeling languages (e.g., [17, p. 320]), although it is to be kept in mind that generality refers to a spectrum, not a dichotomy. Just as 'power', the 'generality' criterion would raise ample complications if it were operationalized.

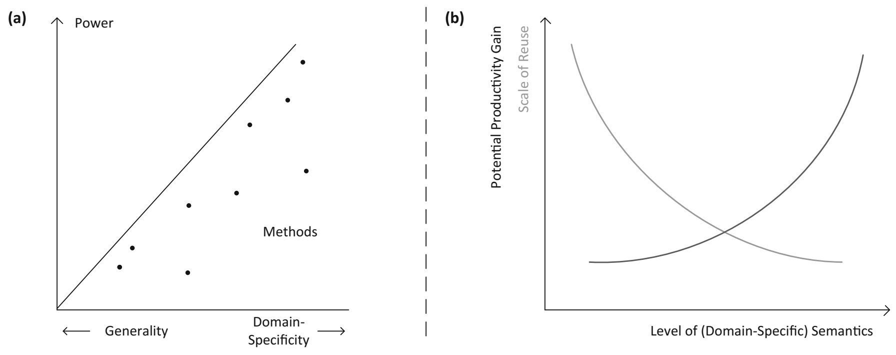


Fig. 1. Representations of the power/generality trade-off (figure (a) adopted and slightly modified from [1, p. 373] (for reasons of consistency, the label 'Domain-Specificity' in figure (a) has been renamed from 'Information demanded' in the original [1, p. 373].); figure (b) adopted from [19, p. 29])

Given these definitions of power and generality, it is readily apparent that the dimensions hold a conflict. The "the more task-independent a method," that is, the *higher the generality* of a method, "the less support it provides for a specific application," that is, the *lower its power* ([3, p. 915]; see also [1, pp. 372–373], [2, p. 606], [4, p. 125], [17, p. 320], [19, p. 29]). Conversely, "the more task-specific a method," that is, the *lower the generality* of a method, "the more useful support it can provide for problems within its range," that is, the *higher its (potential) power* ([3, p. 915]; see also the references above). Part (a) of Fig. 1 illustrates this relationship. With increasing specificity, and decreasing generality, the potential power of a method increases. It is to be noted, though, that power does not *follow* from a low level of generality; some artifacts may not fully harness their specificity [1, p. 373]. Generality thus poses a form of restriction on potential power, represented by the boundary line in part (a) of Fig. 1 [1, p. 373].

What is the reason for the trade-off? As a general rule, artifacts of higher specificity "exert strong[er] expectations on the kinds of tasks and domains in which they can be used" [4, pp. 125–126]. In relation to problem solving methods, this can mean that the employed problem definitions involve more domain-specific concepts and conditions, and that more domain-specific information is required to apply the method [1, pp. 372–373]. Clearly, each domain-specific piece of information "is one more item that can be exploited in finding the solution" [1, p. 372]. In relation to modeling languages, matters can be stated differently. Here the reason can be seen in the *level of semantics* of language constructs (for the following, see [17, p. 320], [19, p. 29], [6, pp. 67–70]). Concepts with a lower level of semantics, such as 'Entity', can be applied to describe almost any domain, but they have little inherent expressiveness. Vice versa, concepts with a higher level of semantics, such as 'General Ledger Account', are more expressive and can be associated with specialized language constraints and visualizations (see [17, p. 320]). But these concepts are limited to narrower application domains. In a treatment of object-oriented design, Graham has thus pointedly concluded that "the fact is that all semantics compromise reuse" [21, p. 299]. Part (b) of Fig. 1 visualizes the relationship between semantics, generality, and power.

2.2 Basic Concepts of Decision and Problem Solving Research

Having summarized the power/generality trade-off, it is useful to consider fundamentals of decision and problem solving research before directing attention at the manifestation of the trade-off in practical decision modeling in Sect. 3. Decision and problem solving research has a long and rich tradition. For purposes of this paper, the discussion will be confined to an examination of basic concepts.

In many fields, including economics and psychology, the notion most commonly associated with the term 'decision' is that of a *choice* (see, e.g., [9, pp. 568–572], [22, pp. 656–657]). In this view, a decision situation consists of a set of *alternative courses of action* (only one of which can be chosen), a set of possible *future states* (whose occurrence is uncertain), a set of *outcomes* attached to the alternatives and future states, and a set of *goals* or criteria that are used to assess the outcomes (see, e.g., [22, pp. 656–657], [23, pp. 806–807]). It will be noted that ordinary modeling languages about routine decisions, such as the OMG DMN, effectively implement this view (cf. [10, pp. 23, 26]).

But the choice view has limitations. It overlooks at least two aspects. First, it does not consider the decision *process* that precedes a "final moment" of choice [24, pp. 39–40], [25, pp. 20–21]. It is long acknowledged that decisions emerge from a process that involves at least (i) an initial recognition of a decision-requiring unclear situation, (ii) a perhaps lengthy analysis of the situation and the identification or construction of possible courses of action, and (iii) the ultimate choice [24, pp. 40–44], [25, pp. 20–29]. The situation analyzed in a decision process is often called *problem* [26, pp. 70–75]. Because decision processes are thus essentially concerned with problems, they are often seen to be closely related to, if not identical with, problem solving processes [26, pp. 70–75]. The extent of problem analysis in a decision process will vary with how structured or ill-structured the situation is taken to be by the decision maker [27, p. 2681]. Importantly, this is a function of the individual. Problems are not *per se* structured or ill-structured; they are viewed to be so by people [28, pp. 324–325]. The second oversight of the choice-centric view results from the previous point: the choice view says nothing about how a representation, or an understanding, of the problem situation is *constructed* by the individual [22, p. 658]. People are not readily given an objective representation of a problem (including goals, alternatives, and so on) [28, pp. 324–325]; instead, they have to construct a personal *mental representation* of the problem [27, p. 2680]. From this it follows that representations of real-world problems are highly dependent on traits of the subject, for example, "previous experience, epistemological beliefs, [and] reasoning skills" [27, p. 2682]. This is especially true for complex problems in organizations. Usually, managers face ambiguous "events that call for evaluation and interpretation" [28, pp. 322], so vastly different problem representations may be build by different actors.

Taken together, the above points also anticipate why the power/generality trade-off is particularly relevant for the *modeling* of decision or problem situations. If "problems are not objective entities in their own right" [28, p. 324], then it is evident that considerable latitude exists in how broadly or narrowly the substance of a problem can be perceived, and, consequently, modeled.

3 Illustration of the Trade-Off in Decision Modeling

Following the theoretical overview, it is instructive to obtain a more tangible view of the power/generality trade-off in decision modeling. In this section, a case study with an existing modeling language will be used to practically examine the conflict and related problems in the traditional conceptual modeling paradigm.

3.1 The Applied Modeling Language

The applied modeling language was developed in prior research [12,29,30]. Its key purpose is to support the documentation, analysis, monitoring, and improvement of decision process types in the organization. The language allows to model decision process types in association with relevant decision premises, involved organizational units, and supporting information systems (IS) (for a case study, see [12, pp. 183–186]). In order to reuse concepts to describe organizational reference elements, the language is designed as part of the larger enterprise modeling method MEMO (Multi-Perspective Enterprise Modeling) [31].

The language has been selected for several reasons. First, in contrast to languages such as the OMG DMN, which solely focuses on "operational decisions made in day-to-day business processes" [10, p. 27], the selected language is intended to model middle ground decision types that are "neither purely routine or rule-based" nor "totally novel" [12, p. 183]. It is assumed that such decision situations are particularly suited to illustrate the power/generality trade-off because they can be defined at many different levels of specificity. Second, unlike many other languages (e.g., [11,13]), the selected approach is explicitly meant to model *types* of organizational decision processes [12, pp. 184, 187]. This focus is helpful for the illustration because the aim to model abstractions (rather than singular decision situations) stresses the need to choose generality/specificity levels for the abstractions. Lastly, in contrast to most other approaches, the selected language is defined using an explicit meta modeling language, MEMO MML [31, pp. 918–920]. MEMO MML is based on the traditional language paradigm and offers meta language concepts to define whether attributes are to be instantiated at the type level M_1 or at the instance level M_0 [31, pp. 918–920]. Attributes of the latter sort are named 'intrinsic features' and marked with a black 'i' in meta models [31, pp. 918–920]. The exactness of MEMO MML is useful because it underlines the mechanisms of the traditional language paradigm.

For present purposes, two language constructs are of primary interest, *'Stimulus'* and *'DecisionProcess'*. The original specification is shown in Fig. 2. The concept *'DecisionProcess'* is the key concept to define types of decision processes

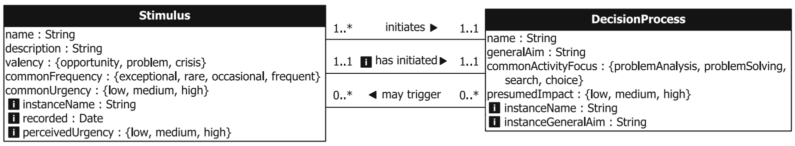

Stimulus
name : String
description : String
valency : {opportunity, problem, crisis}
commonFrequency : {exceptional, rare, occasional, frequent}
commonUrgency : {low, medium, high}
i instanceName : String
i recorded : Date
i perceivedUrgency : {low, medium, high}
1..* initiates ▶ 1..1
1..1 i has initiated ▶ 1..1
0..* ◀ may trigger 0..*
DecisionProcess
name : String
generalAim : String
commonActivityFocus : {problemAnalysis, problemSolving, search, choice}
presumedImpact : {low, medium, high}
i instanceName : String
i instanceGeneralAim : String

Fig. 2. Meta model excerpt from the existing decision process modeling language (adapted from [12, p. 188])

that occur regularly in the organization [12, p. 187]. The concept *'Stimulus'* is meant to model abstractions of events or perceptions that trigger decision processes [12, p. 187]. The idea is that stimuli represent initial problem perceptions, whereas decision processes describe the processes dealing with these problems. As shown in Fig. 2, both concepts hold attributes to be instantiated at the type level M_1 and at the instance level M_0 (marked with an 'i'). Decision processes can be associated with various kinds of reference elements. In the case study below, for example, all modeled decision processes are associated with different types of *'DecisionPremises'*. Decision premises specify the content of a problem. The used subtypes of *'DecisionPremise'* include *'Goal'* (where *'SymbolGoals'* are inspirational goals and *'EngagementGoals'* are measurable goals), *'SituationalAspect'* (a real-world aspect or piece of information deemed relevant for a problem), *'EnvironmentalFactor'* (a real-world factor that affects goal attainment but cannot be controlled), and *'ActionVariable'* (a real-world aspect that affects goal attainment and can be modified) [12, pp. 187–188], [30, pp. 521–524].

3.2 Case Study Illustration and Discussion

Figure 3 shows parts of a model that describes decision processes of a company that produces sweet and savory snacks, distributing them in food market chains and vending machines. On the left-hand side (a) of the model, decision processes with a *high level of generality* are modeled; on the right-hand side (b) of the model, decision processes with a *high level of specificity* are shown. The fact that decision processes of such diversity can be defined is to a large extent a result of the conditions discussed in Sect. 2.2. Problems are not objective entities but individual constructions, and thus they can be perceived more broadly or narrowly. Considering Fig. 3, a number of observations can be made.

The Modeler's Perspective. It is readily apparent that there is an instance of the power/generality trade-off in the example model. The decision process types at the left-hand side (a) are highly general in the sense that they are abstractions of a huge spectrum of vastly different specific processes. For example, the decision process type 'Analyze cause of poor performance and initiate countermeasure' could be instantiated into processes as different as 'Analyze cause of poor brand image and promote it' or 'Analyze cause of poor supply chain throughput time and improve efficiency'. The advantage of this generality is that a single model element covers a great range of real-world events. Less elements have to be defined in an enterprise model, reducing efforts in creating and maintaining the

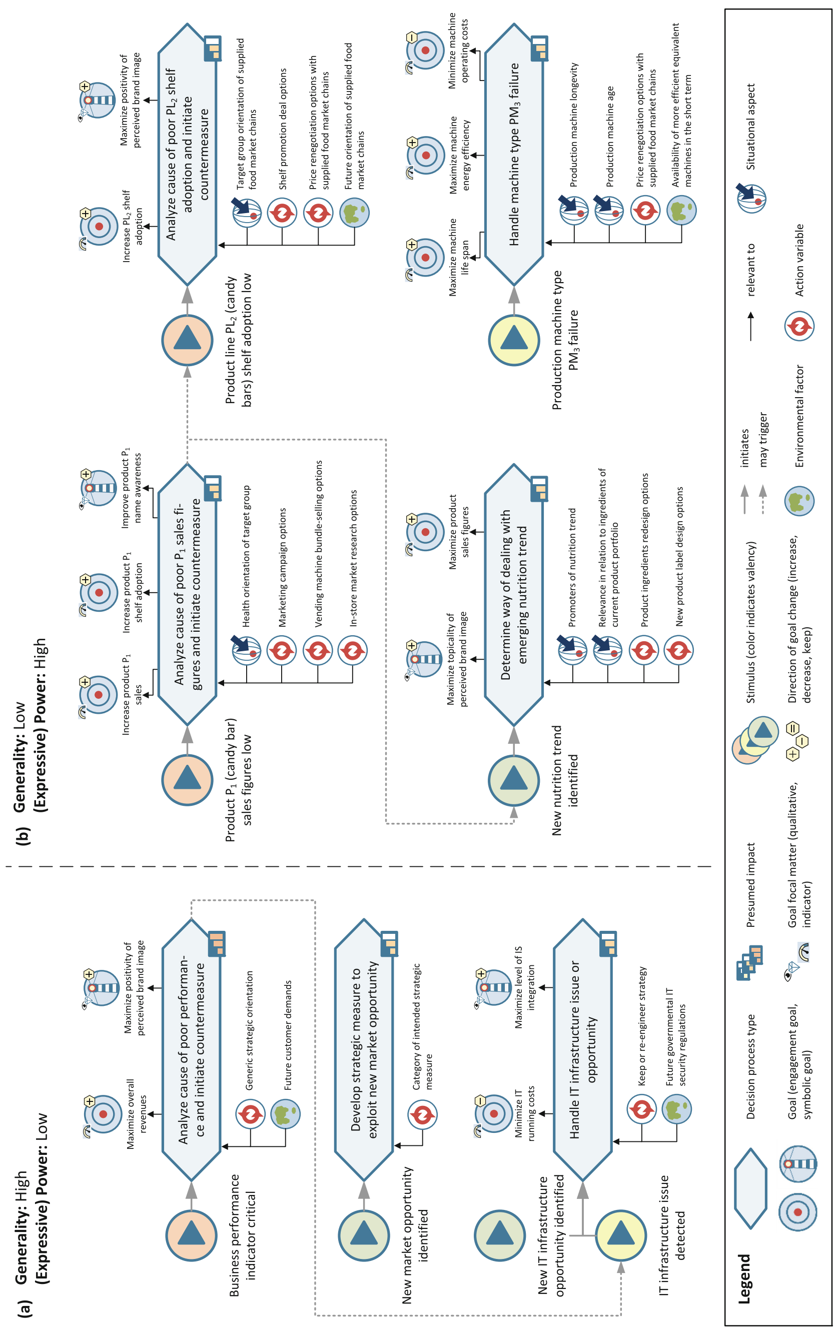

(a) Generality: High (Expressive) Power: Low
Business performance indicator critical
Analyze cause of poor performance and initiate countermeasure
Maximize overall revenues
Maximize positivity of perceived brand image
Generic strategic orientation
Future customer demands
New market opportunity identified
Develop strategic measure to exploit new market opportunity
Category of intended strategic measure
New IT infrastructure opportunity identified
IT infrastructure issue detected
Handle IT infrastructure issue or opportunity
Minimize IT running costs
Maximize level of IS integration
Keep or re-engineer strategy
Future governmental IT security regulations
(b) Generality: Low (Expressive) Power: High
Product P1 (candy bar) sales figures low
Analyze cause of poor P1 sales figures and initiate countermeasure
Increase product P1 sales
Increase product P1 shelf adoption
Improve product P1 name awareness
Health orientation of target group
Marketing campaign options
Vending machine bundle-selling options
In-store market research options
Product line PL2 (candy bars) shelf adoption low
Analyze cause of poor PL2 shelf adoption and initiate countermeasure
Increase PL2 shelf adoption
Maximize positivity of perceived brand image
Target group orientation of supplied food market chains
Shelf promotion deal options
Price renegotiation options with supplied food market chains
Future orientation of supplied food market chains
New nutrition trend identified
Determine way of dealing with emerging nutrition trend
Maximize topicality of perceived brand image
Maximize product sales figures
Promoters of nutrition trend
Relevance in relation to ingredients of current product portfolio
Product ingredients redesign options
New product label design options
Production machine type PM3 failure
Handle machine type PM3 failure
Maximize machine life span
Maximize machine energy efficiency
Minimize machine operating costs
Production machine longevity
Production machine age
Price renegotiation options with supplied food market chains
Availability of more efficient equivalent machines in the short term
Legend
Decision process type
Presumed impact
Stimulus (color indicates valency)
initiates
may trigger
relevant to
Situational aspect
Goal (engagement goal, symbolic goal)
Goal focal matter (qualitative, indicator)
Direction of goal change (increase, decrease, keep)
Environmental factor
Action variable

Fig. 3. Illustration of the power/generality trade-off in decision modeling

model. Another advantage is specific to the modeling of decision processes. If decision process types are defined very broadly, they can be instantiated into future instances that might not have been predicted at the time of modeling. For example, the process type 'Develop strategic measure to exploit new market opportunity' could be instantiated into 'Exploit cooperation with new online grocery retailers', even though this might not have been considered when defining the original type. On the other hand, the disadvantage of overly general decision process types is also apparent. The information and the guidelines that can be documented for such types must remain very broad and vague. For example, the only goals that can be defined for 'Analyze cause of poor performance and initiate countermeasure' are universal goals like 'Maximize overall revenues'.

The other side of the trade-off is visible on the right-hand part (b) of Fig. 3. Here highly specialized decision process types such as 'Analyze cause of poor PL_2 shelf adoption and initiate countermeasure' or 'Handle machine type PM_3 failure' are defined. This has clear advantages. For each special decision process type, it is possible to record highly specific decision premises, providing clear guidance and knowledge for actors in a decision situation. For example, it is possible to note that 'Vending machine bundle-selling options' should be considered as possible courses of action when the sales of a candy product drop. Thus, a high level of specificity affords higher productivity gains when using the model (cf. [17, p. 320]). But the converse is also true. First, the obvious disadvantage of a high level of specificity is that a great number of individual decision process types have to be defined to cover a fair portion of the relevant organizational realm. Furthermore, a high level of specificity makes it more difficult to account for the contingencies of future developments. For one thing, it is almost impossible to enumerate all possible future problems at a high level of detail. For another thing, the exact future problems that will require decisions may simply not be known at model time. It may be obvious, for example, that there will be new market opportunities, but their nature cannot be precisely predicted. Thus, a high level of specificity promotes expressiveness but hampers coverage.

In summary, both general and specific model elements have advantages and disadvantages. In fact, the case study suggests that it is often necessary to combine elements of both kinds, capturing specialized knowledge where necessary and leaving it with broad orientations where sufficient or not possible otherwise.

The Formal Language Perspective. Granted that both general and specific decision process types are needed in the model, several shortcomings of the applied language paradigm can be recognized. To begin, it can be asserted that from a formal perspective there is nothing which explicitly distinguishes the decision process types in the model. All elements in Fig. 3 are instances of the M_2 level meta types shown in Fig. 2, and all of them are located at the type level M_1. This is a consequence of the fact that the traditional conceptual modeling paradigm provides only *one* classification level for the user (e.g., [14, pp. 6–8], [15, pp. 18–20]). The model interpreter has to autonomously reconstruct the order of semantic generality of model elements, which worsens model clarity and interpretability. Furthermore, the model seems to involve hidden, or implicit, instance-of relationships. For example, the specific decision process type 'Analyze cause of poor P_1 sales figures and initiate countermeasure' seems to constitute an *instance* of the more general type 'Analyze cause of poor performance and initiate countermeasure'. If this relationship were made explicit, it would be possible to derive additional information from the model. This is because the features of the more abstract element (such as its attribute values and attached elements) pertain to its instances as well. For example, in the model it could be derived that the general goal 'Maximize positivity of perceived brand image' is relevant to the specific instance as well. Furthermore, the ability to express instance-of relationships would enable more consistent model extensions over time. For example, it would be possible to consistently add a new instance of 'Develop strategic measure to exploit new market opportunity' if a special kind of opportunity becomes important for the company at some point. However, although desirable, instance-of relationships between decision process types of distinct specificity cannot be expressed in the traditional language architecture because, again, only one classification level M_1 is available to the modeler.

In sum, the traditional language paradigm exacerbates the effects of the power/generality conflict. The paradigm entails conceptual ambiguities and leaves unexploited substantial possibilities of abstraction and consistent extension.

4 Multi-level Modeling as a Resolution

It was found in the last section that the modeling of organizational decision processes comes with an instance of the power/generality trade-off. From a modeler's perspective, there is a constant need to choose generality/specificity levels, and usually both levels are required in the model. These difficulties are to some extent an inherent part of applying the considered language, and they cannot be fully resolved in this paper (if at all). However, from a formal language perspective, it was also concluded that the traditional conceptual modeling paradigm further aggravates the difficulties. Unable to express instance-of relationships between elements of different generality/specificity, the paradigm cannot avoid conceptual ambiguities and it ignores significant abstraction potentials. Here additional (semi-)formal support can be provided. Specifically, this paper proposes that the *paradigm of multi-level modeling* provides an architecture within which generality/specificity relationships can be expressed more explicitly and consistently, and that it thereby also offers a remedy for the power/generality trade-off.

This section demonstrates how decision process types of different generality/specificity can be modeled in a multi-level language architecture. In doing so, the section addresses the second purpose of this paper in a twofold way. First, as a concrete contribution, it presents re-conceptualizations of the language constructs of the existing modeling language [12,30] for a multi-level environment. Second, as a general contribution, it showcases how multi-level modeling affords an environment that facilitates dealing with the generality/power trade-off.

4.1 Applied Meta Modeling Architecture

The use of a multi-level environment will be illustrated using the meta modeling language $FMML^x$ [17, pp. 324–327]. This language is selected because unlike other multi-level architectures (e.g., [16]), $FMML^x$ is specifically intended to enable the design of multi-level DSMLs, and it is motivated by the wish to relax the very trade-off that is the topic of the present paper [17, pp. 321–322]. Using $FMML^x$, models can span across any number of abstraction levels, and model elements (concepts, attributes, and relationships) can be instantiated at all of these levels (for the following, see [17, pp. 325–327]). The lowest level (M_0) is the instance level; concepts at higher levels are gradually more abstract. In the concrete syntax of $FMML^x$, white numbers in black boxes attached to model elements define the levels at which these elements are to be instantiated. To improve clarity, the heading of each concept includes its name (shown in a large font size) alongside the name of the concept from which it has been instantiated (shown between two upward arrows). In addition to the original syntax of $FMML^x$, attributes that have been instantiated (i.e., attributes whose value has been set) are marked with white circles below to improve readability.

4.2 Decision Process Models in a Multi-level Environment

Figure 4 shows how the two central concepts of the decision process modeling language, *'DecisionProcess'* and *'Stimulus'*, can be conceptualized in a multi-level environment. The figure displays the abstract syntax of a model (of course, a concrete syntax similar to Fig. 3 could be added here as well). The two core concepts are highlighted by an orange-colored frame. It will be noted that *'Stimulus'* in the most abstract sense is defined at level M_5, whereas *'DecisionProcess'* first appears on M_4. This is because more (sub-)levels of abstraction have appeared useful for *'Stimulus'* than for *'DecisionProcess'* in the given scenario. A number of attributes have been defined for both concepts. Some attributes will be recognized from the original specification (Fig. 2), while others have been newly introduced to exploit the extended modeling space provided by multi-level modeling. As is expressed by the numbers in the black boxes next to the attributes, the attributes are to be instantiated at different levels from M_4 to M_0. Also, the already known relationship 'initiates' reappears, but now it can be instantiated at different levels as well. In order to provide an informative picture of how the proposed conceptualization can be used, Fig. 4 shows a sizable number of instances of both core concepts. The modeled scenario describes decision processes from the same snack food company that has already been portrayed in Fig. 3. Going down from the highly general concepts at level M_5 or M_4, increasingly specific instances are defined, and successively more attributes are instantiated in the process. Importantly, the fact that attribute values are *instantiated* between abstraction levels also demonstrates that traditional generalization/specialization relationships on a single classification level could *not* be used to the same effect (cf. [17, p. 323]). Keeping in mind the conclusions

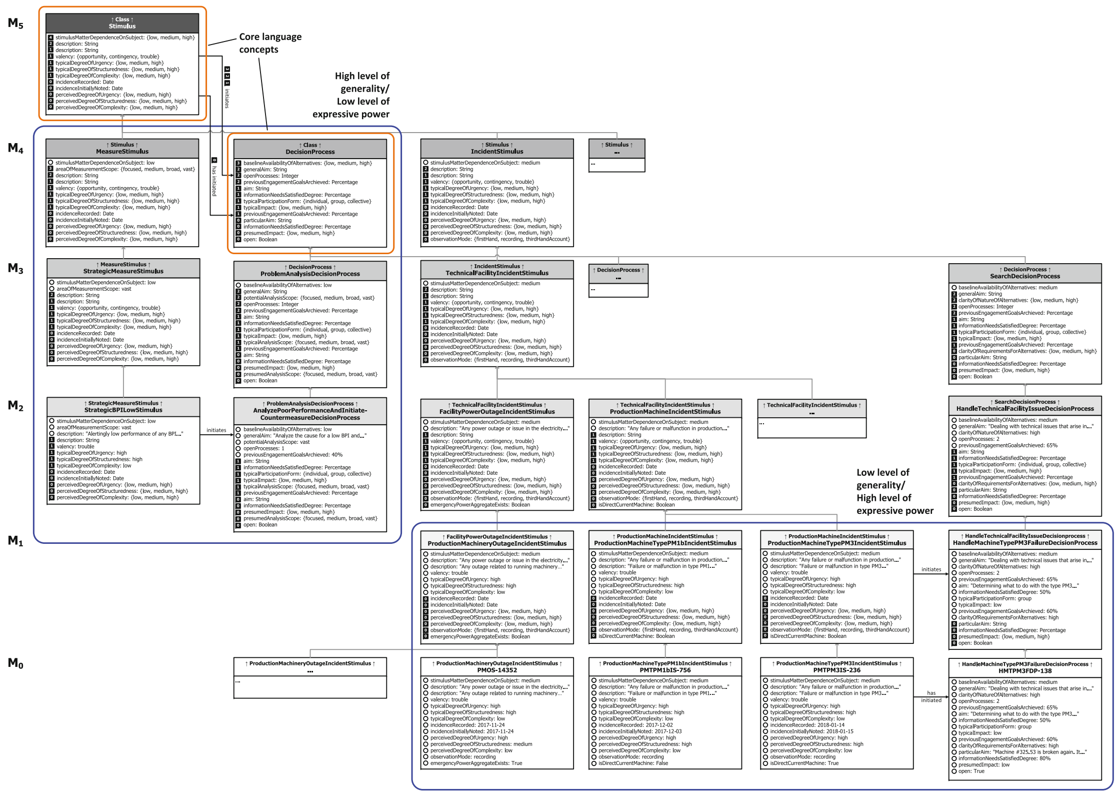

M5
M4
M3
M2
M1
M0
Core language concepts
High level of generality/ Low level of expressive power
Low level of generality/ High level of expressive power
↑ Class ↑ Stimulus
↑ Class ↑ DecisionProcess
initiates
has initiated
↑ Stimulus ↑ MeasureStimulus
↑ Stimulus ↑ IncidentStimulus
↑ MeasureStimulus ↑ StrategicMeasureStimulus
↑ DecisionProcess ↑ ProblemAnalysisDecisionProcess
↑ IncidentStimulus ↑ TechnicalFacilityIncidentStimulus
↑ DecisionProcess ↑ SearchDecisionProcess
↑ StrategicMeasureStimulus ↑ StrategicBPILowStimulus
↑ ProblemAnalysisDecisionProcess ↑ AnalyzePoorPerformanceAndInitiate-CountermeasureDecisionProcess
↑ TechnicalFacilityIncidentStimulus ↑ FacilityPowerOutageIncidentStimulus
↑ TechnicalFacilityIncidentStimulus ↑ ProductionMachineIncidentStimulus
↑ SearchDecisionProcess ↑ HandleTechnicalFacilityIssueDecisionProcess
↑ FacilityPowerOutageIncidentStimulus ↑ ProductionMachineryOutageIncidentStimulus
↑ ProductionMachineIncidentStimulus ↑ ProductionMachineTypePM1bIncidentStimulus
↑ ProductionMachineIncidentStimulus ↑ ProductionMachineTypePM3IncidentStimulus
↑ HandleTechnicalFacilityIssueDecisionprocess ↑ HandleMachineTypePM3FailureDecisionProcess
↑ ProductionMachineryOutageIncidentStimulus ↑ PMOS-14352
↑ ProductionMachineTypePM1bIncidentStimulus ↑ PMTPM1bIS-756
↑ ProductionMachineTypePM3IncidentStimulus ↑ PMTPM3IS-236
↑ HandleMachineTypePM3FailureDecisionProcess ↑ HMTPM3FDP-138

Fig. 4. Re-conceptualization and example instantiations of decision process modeling constructs in a multi-level language architecture (Color figure online)

reached in Sect. 3.2, several benefits of the multi-level modeling paradigm can be highlighted.

First, and most basically, the model in Fig. 4 *explicitly distinguishes* concepts at different levels of generality/specificity. In contrast to a model in the traditional paradigm, it can formally be derived from the model that, for example, the stimulus 'Production Machine Type PM3 Incident' at level M_1 is more specific than the stimulus 'Technical Facility Incident' at level M_3. Second, the model specifies *explicit instance-of relationships* between different concepts, enabling to derive additional information when interpreting specific elements. For example, the decision process type 'Handle Machine Type PM3 Failure' at level M_1 is now an explicit instance of the process type 'Handle Technical Facility Issue' at level M_2. This enables to inspect attribute values (such as for 'typicalDegreeOfUrgency') that have already been instantiated at higher levels. Furthermore, it enables to acquire additional information by tracing the instance-of relationships from specific to more general elements and to inspect associations with other concepts defined at these higher level. Third, and this is a consequence of the former two points, the model permits to *combine generic and specific model elements* in a more structured and elegant fashion. As is highlighted in the blue-colored frames in Fig. 4, it is possible for certain branches in the concept hierarchy to remain at a rather high level of generality (see the left-hand blue frame), whereas other elements may be detailed at a high level of specificity (see the right-hand blue frame). Specifically, the company has decided to keep the model part about low performance indicators at a general level, while the model part about facility incidents has been elaborated in great detail. This choice may be the result of perceived importance or availability of information. As a whole, the approach offers an accessible and orderly way of combining different specificity levels. Fourth, the multi-level model affords *more consistent gradual extensions*. For example, if the company at some point decides to define specialized decision process types to handle particular performance figures, it would be possible to add them coherently and consistently as new instances in the left-hand branch of Fig. 4. Lastly, and this corresponds to the original motivation of the $FMML^x$ (see [17, pp. 320–322]), with a multi-level architecture the distribution and development of DSMLs can be organized so that it yields additional economies of scale. Specifically, it would be possible for professional language designers to focus on concepts at higher levels of abstractions (e.g., M_5 to M_3) and have companies define their own "local dialects" at lower levels to incorporate industry- or company-specific needs [17, p. 321]. This would help relax the power/generality trade-off even in the design and dissemination of DSMLs.

To conclude, although the power/generality trade-off remains a substantial issue that modelers (especially, but not only) in the realm of decision modeling must deal with, it was shown that the use of a multi-level language architecture offers a numbers of benefits remedying its implications. A multi-level paradigm contributes to formal discriminability, consistent extensibility, structured organization, and even the design and distribution of decision models and DSMLs.

5 Conclusions

This paper has made two contributions, both on an abstract and a concrete level. First, the paper has directed attention at the power/generality trade-off in decision and problem modeling, and it has practically illustrated that conflict in relation to an existing modeling language. Second, the paper has argued that the use of a multi-level language paradigm provides several benefits to alleviate the implications of the conflict, and it has presented concrete re-conceptualizations of an existing modeling language to demonstrate these advantages.

The arguments proposed in this paper, however, prompt a number of further research activities to become actionable. First, although the paper has theoretically reviewed the power/generality trade-off, ramifications of that trade-off in conceptual modeling have mostly been explored by means of examples. In future research, a more systematic examination of the formal and conceptual repercussions of the conflict needs to be conducted. Following this, the proposed resolution through conceptual multi-level modeling should be systematically compared with the use of other modeling paradigms and abstraction concepts (such as traditional generalization/specialization relationships in classical language architectures). Second, while this paper has focused on the power/generality trade-off in decision and problem modeling, the conflict exists in other domains as well. For example, when modeling enterprise risks (e.g., [32]) the modeler constantly has to choose how broadly or sharply risks should be delineated. It is an interesting question for future research whether such other domains yield different conditions for generality/specificity choices. Lastly, the new conceptualizations for the decision process modeling language presented in this paper are not yet mature and complete. Future research needs to further refine and evaluate these concepts, and to extend them with other constructs so as provide a coherent multi-level decision process modeling language.

References

1. Newell, A.: Heuristic programming: ill-structured problems. In: Aronofsky, J.S. (ed.) Progress in Operations Research, vol. III, pp. 361–414. Wiley, New York (1969)
2. de Mast, J., Lokkerbol, J.: An analysis of the Six Sigma DMAIC method from the perspective of problem solving. Int. J. Prod. Econ. **139**(2), 604–614 (2012)
3. Fensel, D., Motta, E.: Structured development of problem solving methods. IEEE Trans. Knowl. Data Eng. **13**(6), 913–932 (2001)
4. Klinker, G., Bhola, C., Dallemagne, G., Marques, D., McDermott, J.: Usable and reusable programming constructs. Knowl. Acquis. **3**(2), 117–135 (1991)
5. Beys, P., Benjamins, V.R., van Heijst, G.: Remedying the reusability-usability tradeoff for problem-solving methods. In: Proceedings of the 10th Workshop on Knowledge Acquisition for Knowledge-Based Systems (KAW96) (1996)
6. Frank, U.: Multiperspektivische Unternehmensmodellierung: Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung. Oldenbourg, München (1994)

7. Becker, J., Delfmann, P., Knackstedt, R.: Adaptive reference modeling: integrating configurative and generic adaptation techniques for information models. In: Becker, J., Delfmann, P. (eds.) Reference Modeling, pp. 27–58. Physica, Heidelberg (2007)
8. Winter, R.: Design solution analysis for the construction of situational design methods. In: Ralyté, J., Mirbel, I., Deneckère, R. (eds.) ME 2011. IAICT, vol. 351, pp. 19–33. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19997-4_4
9. Goldstein, W.M., Weber, E.U.: Content and discontent: indications and implications of domain specificity in preferential decision making. In: Goldstein, W.M., Hogarth, R.M. (eds.) Research on Judgment and Decision Making, pp. 566–617. Cambridge University Press, Cambridge and New York (1997)
10. Object Management Group: Decision model and notation: Beta 1. OMG Document dtc/2014-02-01 (2014)
11. Plataniotis, G., de Kinderen, S., Proper, H.A.: EA Anamnesis: an approach for decision making analysis in enterprise architecture. Int. J. Inf. Syst. Model. Des. **5**(3), 75–95 (2014)
12. Bock, A.: Beyond narrow decision models: toward integrative models of organizational decision processes. In: 17th IEEE Conference on Business Informatics (CBI 2015), pp. 181–190. IEEE Computer Society (2015)
13. Horkoff, J., Barone, D., Jiang, L., Yu, E.S., Amyot, D., Borgida, A., Mylopoulos, J.: Strategic business modeling: representation and reasoning. Softw. Syst. Model. **13**(3), 1015–1041 (2014)
14. Fill, H.G., Karagiannis, D.: On the conceptualisation of modelling methods using the ADOxx meta modelling platform. Enterp. Model. Inf. Syst. Archit. **8**(1), 4–25 (2013)
15. Object Management Group: OMG Unified Modeling Language (OMG UML), Infrastructure: Version 2.4.1. OMG Document formal/2011-08-05
16. Atkinson, C., Gutheil, M., Kennel, B.: A flexible infrastructure for multilevel language engineering. IEEE Trans. Softw. Eng. **35**(6), 742–755 (2009)
17. Frank, U.: Multilevel modeling: toward a new paradigm of conceptual modeling and information systems design. Bus. Inf. Syst. Eng. **6**(6), 319–337 (2014)
18. Carvalho, V.A., Almeida, J.P.A.: Toward a well-founded theory for multi-level conceptual modeling. Softw. Syst. Model. **17**(1), 205–231 (2018)
19. Frank, U.: Enterprise modelling: the next steps. Enterp. Model. Inf. Syst. Archit. **9**(1), 22–37 (2014)
20. Chen, P.P.S.: The entity-relationship model–toward a unified view of data. ACM Trans. Database Syst. **1**(1), 9–36 (1976)
21. Graham, I.: Object Oriented Methods, 2nd edn. Addison-Wesley, Wokingham (1994)
22. Hastie, R.: Problems for judgment and decision making. Annu. Rev. Psychol. **52**, 653–683 (2001)
23. Keeney, R.L.: Decision analysis: an overview. Oper. Res. **30**(5), 803–838 (1982)
24. Simon, H.A.: The New Science of Management Decision, Revised edn. Prentice-Hall, Englewood Cliffs (1977)
25. Lundberg, C.C.: Administrative decisions: a scheme for analysis. In: Gore, W.J., Dyson, J.W. (eds.) The Making of Decisions, pp. 17–30. Free Press, New York (1964)
26. Kirsch, W.: Entscheidungsprozesse: Erster Band: Verhaltenswissenschaftliche Ansätze der Entscheidungstheorie. Gabler, Wiesbaden (1970)
27. Jonassen, D.H., Hung, W.: Problem solving. In: Seel, N.M. (ed.) Encyclopedia of the Sciences of Learning, pp. 2680–2683. Springer, New York (2012). https://doi.org/10.1007/978-1-4419-1428-6_208

28. Dery, D.: Decision-making, problem-solving and organizational learning. Omega **11**(4), 321–328 (1983)
29. Bock, A., Kattenstroth, H., Overbeek, S.J.: Towards a modeling method for supporting the management of organizational decision processes. In: Fill, H.G., Karagiannis, D., Reimer, U. (eds.) Modellierung 2014. Lecture Notes in Informatics, vol. P-225, pp. 49–64. Gesellschaft für Informatik, Bonn (2014)
30. Bock, A., Frank, U.: MEMO GoalML: a context-enriched modeling language to support reflective organizational goal planning and decision processes. In: Comyn-Wattiau, I., Tanaka, K., Song, I.-Y., Yamamoto, S., Saeki, M. (eds.) ER 2016. LNCS, vol. 9974, pp. 515–529. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46397-1_40
31. Frank, U.: Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges. Softw. Syst. Model. **13**(3), 941–962 (2014)
32. Strecker, S., Heise, D., Frank, U.: RiskM: a multi-perspective modeling method for it risk assessment. Inf. Syst. Front. **13**(4), 595–611 (2011)

Modeling Organizational Structures in the Realm of Enterprise Modeling: Limitations of the Current Paradigm and Prospects of Multilevel Language Architectures

Sybren de Kinderen(✉) and Monika Kaczmarek-Heß

University of Duisburg-Essen, Universitätsstraße 9, 45141 Essen, Germany
{sybren.dekinderen,monika.kaczmarek}@uni-due.de

Abstract. Given the notable role that organizational structures play in different analyses, modeling constructs for organizational structures are present in a number of enterprise modeling approaches. In this paper, we point out limitations of modeling approaches for expressing organizational structure, especially with regards to (1) expressing domain characteristics, and (2) suitability of current modeling approaches for addressing typical analysis questions. As a response, we discuss how a promising novel modeling paradigm, called multilevel modeling, can be used to address these limitations. We use the organizational structure of universities as a case scenario.

Keywords: Multilevel modeling · Organizational structure modeling FMML^{x}

1 Introduction

Enterprise modeling (EM) is widely accepted as an instrument to support sense-making of an enterprise's action system and information system [1,2]. Enterprise modeling approaches usually cover multiple perspectives on an organization (e.g., by considering in tandem organizational goals, business processes, or IT infrastructure), and relate these perspectives to each other.

An organizational structure is one aspect that EM focuses on. In line with [3], an organizational structure can be defined as "an abstraction of organisation that combines institutional and instrumental aspects. It consists of organisational units (institutional aspect) and their relationships (line of command, responsibilities...) which stresses an instrumental view" (pp. 6–7). The organizational structure of an enterprise plays a pivotal role in various analysis scenarios. For instance, managerial analysis questions focus on authority and responsibility relations between organizational units and/or roles within an enterprise, cf. [4]. Indeed, the model of organizational structure, similar to an organizational chart, can be useful in understanding the basic division of work, cf., [3,4]. It shows various parts of an organization (e.g., positions, or departments), and how these parts are interrelated [4]. Furthermore, when populated with a data-set concerning the running organization, organizational structure models can help to answer analysis questions like "What is the size of the faculty of medicine, in terms of scientific personnel?".

Given the notable role that organizational structures play in different analyses, modeling constructs for organizational structures are present in a number of enterprise modeling frameworks and languages. Exemplary approaches include ArchiMate [5], Architecture of Integrated Information Systems (ARIS) [6,7], 4EM [8], and Multi-Perspective Enterprise Modeling (MEMO) [9]. However, current approaches possess notable limitations when it comes to modeling and analyzing organizational structures. As explained further in detail in Sect. 3.2 such limitations include, among others, (1) not accounting for domain-specific hierarchies of organizational structures spanning multiple classification levels, cf., [10], (2) lacking balance between model reuse and model productivity, meaning that one can either include semantically rich concepts in the language (e.g., to provide an elaborate conception of the concept "Legal Department"), or abstract concepts that can be reused across different languages (e.g., an abstract concept of "Organizational Unit"), but not both at the same time, cf. [11]; (3) not keeping the model and the data of the running organization in sync, thus inhibiting using the approach for analysis questions like the current size of different university faculties.

As we argue in this paper, a major underlying cause of these limitations lies in the traditional two-level modeling paradigm that is used by current modeling approaches together with limitations of the dominant object-oriented programming languages used to implement supporting tools. As a response, in this paper we present the results of an ongoing research on the modeling of organizational structures using an alternative language architecture. We use a case scenario of modeling organizational structure of universities to illustrate limitations of existing approaches and show prospects of multilevel modeling.

The paper is structured as follows. First, a short overview on understanding of organizational structures, basic features of conventional meta modeling, and existing approaches to model organizational structures is given. In Sect. 3 we present a scenario focusing on modeling organizational structures of universities and then, use it to discuss problems and challenges encountered while trying to model it using a traditional modeling approach. Next, we introduce the notion of multilevel modeling and discuss shortly prospects resulting out of its application using a multilevel model of organizational structures of universities. The paper concludes with final remarks and an outlook on future work.

2 Background

2.1 Organizational Structures

An organizational structure refers to the formal configuration between individuals and groups, which defines the allocation of tasks, responsibilities, and authority within the organization [12]. Thus, organizational structures consider (at least implicitly) a wide variety of aspects, such as authority, communication, delegation, responsibility, control, and power [12]. These aspects pertain to both a vertical dimension, such as the line of reporting and command as present in a hierarchy, and a horizontal dimension, such as the pattern of interactions among organizational employees [4, p. 56]. However, there is an inherent tension between the horizontal and vertical structure. Aspects covering the vertical dimension are designed for control, whereas horizontal ones are designed for coordination and collaboration, which usually means reducing control [4, p. 58]. Regarding the latter, communication relations allow the definition of interactions between business actors without requiring the establishment of authority [4,13], whereby "authority" is understood as the right to command and the power to be obeyed [4,13].

Division of labor, meanwhile, describes how work is divided/assigned to different organizational units, cf. [3]. It follows that, e.g., an overall mission is divided into specialized goals or tasks, which are in turn allocated to defined units of work in order to increase efficiency. Indeed, [13] argues that the division of labor aims to produce more (or better) outputs with the same effort by reducing the number of objectives an organizational unit needs to pay attention and effort to.

Organizational units come in different varieties. Indeed, different structuring principles (e.g., functional, line-staff, divisional, matrix and flat organizations) lead to different types of structuring units like departments, divisions, line units, staff units, teams and task forces [4]. The differentiation of organizational units can be reflected in the design of an organizational structure, depending on the required levels of abstraction. It can range from an abstract overview, as one finds it in typical organizational charts which typically focus on hierarchical and staff relations only, to detailed descriptions of jobs and positions, in relation to different organizational roles that need to be fulfilled [4], or a grouping of roles based on similarity of background and expertise, and thus, similar domain conceptualizations [14].

2.2 Conventional (meta) Modeling

In conventional meta modeling, one typically defines the domain concepts and their relations using a meta model (M_2) (language specification). A meta model defines the abstract syntax and semantics of a given language together with additional constraints [15, p. 3]. Being a model of a model, this meta model can subsequently be instantiated on the type level (M_1) (language application). Finally, this type level can be instantiated to the instance level (M_0). The Meta Object Facility (MOF) is the prototypical example of a language architecture following conventional modeling, but many other (enterprise) modeling approaches follow it likewise. Such a conventional (meta) modeling comes with a set of rules [11,16], among others, (1) a type/instance dichotomy, implying, among others, that (meta) types cannot have a state, (2) a strict separation between different levels (i.e., between the language specification and application) meaning that the only relation allowed between objects from different levels (e.g., M_2 and M_1 or M_1 and M_0) is the instantiation relation, and (3) no possibility to account on M_2 for information applicable on M_0.

Corresponding modeling tools are usually developed using mainstream object-oriented programming languages. As those languages feature only one classification level, therefore types or even meta types are represented as objects by overloading the M_0 level of a programming language [17]. Thus, a common representation of code and model is not possible and a model-code synchronization is required [17]. Therefore, not only a recompilation step of modeling tools is required whenever we want to change something in the language specification, which negatively impacts language extensions and modifications possibilities, but also equipping model elements with operations is hardly conceivable [16].

2.3 Modeling of Organizational Structures with EM

Various EM approaches provide modeling concepts to account for organizational structure. For instance, ArchiMate provides concepts such as *Business Actor* and *Business Role*. In the ArchiMate specification, a *Business Actor* is defined as "an organizational entity that is capable of performing behavior" [5] and can express an individual entity or a group entity (e.g., a "Department"). In turn, a *Business Role* (e.g., "Project Manager") is defined as "responsibility for performing specific behavior, to which an actor can be assigned" [5]. A *Business Role* can be assigned to a *Business Actor* through the relationship "assignment". The concept *Business Collaboration* (e.g., a "Supply Chain") represents the interactions between two or more Business Roles. Nevertheless a notable limitation is that ArchiMate offers limited modeling productivity for modeling organizational structures. By this we mean that, by design, ArchiMate offers only limited expressiveness of modeling concepts in terms of, e.g., attributes or constraints. For instance, natively it offers no additional constructs to express hierarchy, or a differentiation between different forms of organizations (functional organizations, line organizations, matrix organizations, etc.), cf. also [18].

ARIS, another EM approach, allows to model types of roles and types organizational units (i.e., it offers modeling constructs such as *Position Type*, *Organization Unit Type* and *Person Type*). However, in most cases, it does not allow to account for further hierarchies. For instance whereas it is possible to define a specialization of *Person Type*, it is not possible to define further specializations of organizational unit types. In addition, the presented concepts lack a clear semantic foundation [19], which has been identified while using the Unified Foundational Ontology to study semantics of ARIS's concepts [20].

MEMO OrgML for structures [3] offers a rich set of elements, which include, among others, (1) positions (and their specializations), (2) organizational units with a rich semantic characteristics allowing to express e.g., larger organization unit, smaller organization unit, or staff unit, (3) committee, and (4) board (e.g., board of directors). In contrast to other approaches, the semantics of the elements and relationships are well described. This is reflected in a variety of relationships types, including "composed of", "supervised by", and "subordinated to". In addition to describing organizational structures, the language permits to record a number of detailed properties for all elements. This includes attributes to describe the position type ("Sales", "Technical", and others), "averagePerformance", or the "requiredQualification", cf. [3].

In 4EM [1] organizational structures are captured by defining types of actors and resources involved in enterprise activities. Actors and resources can be individuals (e.g., a person in the enterprise), organizational units, nonhuman resources, and roles. Individuals may play roles and belong to organizational units as well as be related other elements [1]. Organizational unit can represent any organizational structure in an enterprise [1]. As organizational units are actors they can have sub-units, they may play roles and have other actors belonging to them [1]. Roles can be generalized or specialized, and be component roles. Roles may perform processes and be responsible for performing processes and achieving of goals [1]. Binary relationships are used for describing different kinds of relationships between its components: responsibility and dependency. So, similarly to ArchiMate, 4EM provides modeling concepts to capture instance-level notions of organizational structures. However, although it offers semantically richer concepts than ArchiMate, it does not account for type-level data. Also the semantics of 4EM do not possess the level of detail of MEMO OrgML.

3 Modeling an Organizational Structure

3.1 Scenario: Modeling Organizational Structures of Universities

In order to illustrate challenges related to the modeling of organizational structures, we focus on developing a language of organizational structures of universities. Such as language can (1) foster, e.g., among freshmen students, an understanding of a university itself (e.g., when it was founded) and of its organizational structure (e.g., who is its dean); (2) act as an analysis-tool for the needs of reporting and controlling, so as to help answer analysis questions like how large the faculties are, or how many professorships and different positions belong to a given faculty. In addition, the modeling language should provide a rich expressiveness of the organizational structure of universities (e.g., concepts specific for the higher education institutions). However, at the same time, the language should support reuse in the sense that the language should be applicable to all types of universities world-wide.

The organizational structures of universities vary depending on institutional type, culture, and country, however they also have much in common. Universities incorporate key authority structures, including a rector and chancellor with a cohort of administrative leaders. In this scenario we focus on the structure of German universities.

Education in Germany is not centrally regulated. Each of 16 states is permitted to issue its own university regulations and guidelines. Also, German universities have a great deal of independence. A university consists usually of a number of faculties, which in turn are divided into departments (cf. Fig. 1). Each faculty within a university is under the direction of a dean. A chair person or department head supervises individual departments/research groups. Faculty members are ranked, in descending order, as professor, associate professor, assistant professor, and research assistants.

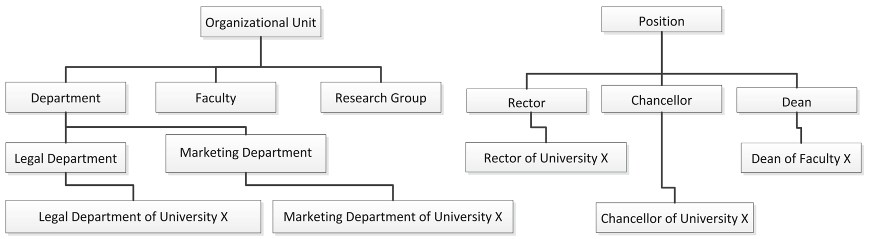


Fig. 1. Excerpts: exemplary hierarchies of organizational structure concepts

Internal university administration is composed of two interrelated administrative cohorts: one is responsible for the oversight and administration of academic affairs; the other is charged with institutional administration. Higher education institutions are governed by a full-time head, a rector. The rector provides overall leadership to the institution and presides over its academic and administrative bureaus.

Within the academic administration, the rector presides over a hierarchy that consists of a number of senior officers, including deans of individual faculties. Academic administrators are drawn from the faculty ranks. Concerning a particular instance: at the University Duisburg-Essen[1], we have the following organizational structure (note that, due to space constraints, we focus mostly on those parts that are relevant for the purposes of this paper). The university is lead by a Rectorate. The Rectorate consists, among others, of the Rector and the Vice-Rectors for Research. The Rectorate is head of the various faculties of the university, among others, the Faculty of Humanities, of Social Sciences, or Economics and Business Administration. Members of a Faculty are research groups. A research group is led by a professor; research staff and student staff belong to a research group.

3.2 Limitations of the Current Paradigm

Now, if we would like to model the university domain using one of the existing enterprise modeling languages, we notice that modeling elements which they provide do not allow us to account for the specificity of university domain. We could be using the general modeling constructs like *Organizational Unit* and, each time we want to model organizational structure of a university, reconstruct necessary concepts on the type level, however, this would substantially hinder the modeling productivity and lead to redundancy. In addition, none of the existing

[1] http://www.uni-due.de.

approaches provides the necessary support for the desired analyses. Therefore, the definition of a domain-specific modeling language (DSML) seems to be reasonable. In this way, we may provide semantically-rich domain specific concepts in the language specification that could be used to model (language application) organizational structure of universities as demanded by the scenario description. However, we observe the following limitations when it comes to employing a traditional language architecture for creating a DSML for expressing and subsequently analyzing the organizational structure of universities.

Limitation: Deciding between language specification and language application. *Rationale:* Finding concepts suitable for a modeling language specification is a challenge in its own right. Next to identifying domain concepts and their typical semantics, requiring extensive literature synthesis, this challenge involves deciding on adopting the concept into the DSML specification at hand by using criteria such as, for a given candidate concept, cf. [21]: "are the semantics of the candidate concept invariant over time?", "does the concept have differentiating characteristics", and "does the candidate concept have a relevance for the analysis scenarios to be supported by the language?". Making such decisions on the language specification versus application is a non-trivial, and thus time-consuming, and constituting a substantial effort of the language designer. Moreover, having to make this decision (which concepts should be part of language application and which of language specification) in the first place is often not satisfactory, as it usually involves a compromise (e.g., deciding between the concept's relevance, invariant semantics, differentiating characteristics).

In addition, as a conceptual model is intended to serve as an abstraction that is independent from states or changes of particular instances, conceptual modeling usually aims at representing types. However, although models of organization structures are abstractions of organizational reality, they actually represent instances of organizational units such as a particular legal department unit of some university, cf. Fig. 1. It creates yet another problem for specifying the language. *Example:* Consider the concepts *Organizational Unit*, versus *Department*, and *Legal Department*, cf. Fig. 1. When designing our DSML we have to invest into at least roughly specifying said concepts, so that we can decide if they will be part of our DSML, or if the concepts will be expressed through the language. This is a non-trivial effort, involving different trade-offs, even when following the guidelines proposed in [21]. For example, while a Department is definitely relevant for analysis purposes, and therefore, one could argue that should be part of the DSML, there are no characteristics that substantially set a *Department* apart from an *Organizational Unit.* Furthermore, while having similar semantics, *Organizational Unit* exhibits a higher level of abstraction, thus it can be used beyond expressing the particularities of departments.

Limitation: Lacking support for reuse of organizational concepts or lacking support for productivity of modeling and analysis.
Rationale: While designing a language following traditional language architectures we need to balance modeling productivity and modeling reuse. This means that we either need to decide to reconstruct a domain with semantically rich

domain concepts (model productivity) so as to equip the modeling language with a rich expressiveness, or to remain on abstract level in one's language specification so that the concepts can be reused in numerous scenarios (model reuse). One cannot have both, cf. [11]. However, having model reuse and model productivity at the same time would allow us to sharpen the semantics of modeling concepts by application over time in different modeling settings, and by application to different modeling scenarios. *Example: Organizational Unit* is not part of the university domain only but is valid for all organizations as its various (reflexive) relationships express division of work, which is fundamental to any organization [3]. Thus, if we decide to equip the concept *OrganizationalUnit* with a set of attributes relevant to our domain (e.g., *noOfFullProfessors*), we would support modeling productivity and facilitate analysis in our domain, but would hamper the range of reuse of the concept by making it not applicable in other domains.

Limitation: Accounting for multiple classification levels as well as accounting for domain values already within a language specification.
Rationale: In language architectures based on the traditional modeling paradigm, we have a limited amount of classification levels at our disposal. Yet, a non-trivial domain modeling task usually involves making decisions on expressing model elements at multiple levels of abstraction. For example, recall our discussion on the exemplary university hierarchy *Organizational Unit*, *Department*, *Legal Department*, a particular *Legal Department* (cf. Fig. 1). While traditional language architectures offer workarounds for expressing such different levels of abstraction, these workarounds are often problematic in that (1) we still have to overload one classification level with multiple abstraction levels, which leads to so called accidental complexity of a model [22], (2) due to the type/instance dichotomy, we need to compromise between the instantiation relation and the specialization relation. Indeed, the refinement relation within organizational structure in some respects represents a specialisation relationship, e.g. *Department* is a specialization of *Organizational Unit*. However, in some respects the refinement relationship represents an instantiation relationship, e.g. *Department* is an instance of *Organizational Unit*. *Example:* A university is staffed according to role types such as *Professor* or *Dean* and at the same time according to different types of organizational units (e.g., *Faculty*, *Department*), each of which may impose constraints on some required role types (e.g., each *Faculty* has a *Dean*). Thus, to describe this domain, one needs to represent entities of different classification levels, e.g., individual persons (e.g., *John Smith*), roles (e.g., *Dean of Faculty of Economics at the University Duisburg Essen*), role types (*Dean*), organizational units (*Research Group of Enterprise Modeling at University Duisburg-Essen*) and organizational unit types (*Research Group*) and its type of a type (*Functional Organizational Unit* of type *Organizational Unit*). Please note that the refinement relation between those concepts is not always as clear: moving down the hierarchy the concepts' structure is extended with new features (specialization relationship), but at the same time values of some characteristics become known (instantiation relationship). If we decide to account for this hierarchy

within the language specification, we would need to squeeze the above concepts into one level of classification. Thus, we would be limited to defining a specialization relationship between those concepts. However, this would not allow us to account for domain constraints and dependencies, nor can we account for the already known values of some properties. Moreover the relevant information should then always be provided on the instance level only, which leads to redundancy.

Limitation: Providing support for automated analysis and calculations.
Rationale: In EM approaches that follow the traditional paradigm, a (meta) model exists separately from running data. This inhibits keeping model and data in sync, which consequently limits the use of a (meta) model for analysis purposes of a running organization. *Example:* Since (cf. Sect. 2.2) the meta model of the university is to be used for controlling and analysis purposes, a closeness to data of the running university is desired to support analysis questions like: *"What is the size of the faculty of medicine, in terms of scientific personnel?"*, or *"What is the size of the different research groups of the faculty of economics?"*. As already mentioned, due to the fact that the currently available modeling tools are usually based on semantics of mainstream object-oriented programming languages, a common representation of model and code is not possible and thus, equipping model elements with operations or linking them to operational level data is hardly conceivable.

4 Multilevel Modeling

In the light of the discussion in Sect. 3.2, the application of conventional two-level modeling paradigm imposes important limitations, which hinder us from delivering a satisfactory solution for modeling organizational structures. Indeed, in order to account for peculiarities of organizational structures, we need a language architecture that deviates from the rules of conventional meta modeling as described in Sect. 2. Firstly, we need support for multiple classification levels, as having only two levels of classification would force us to 'squeeze' the hierarchy of organizational structure-specific concepts into one layer and thus, overload it with different levels of abstraction, cf. [11,22]. Also, to further support abstraction it would be beneficial to rely on deferred instantiation. Given a classification level, deferred instantiation enables one to define on this classification level what we know would be relevant on not directly proceeding lower levels of classification. This way, we increase the reuse of the modeling concepts. For example, while the abstract concept of an *Organization* will have a hierarchy of concepts that instantiate it (e.g., into a University), we know already that a particular Organization will have a specific mission. Thus, we can use deferred instantiation to define for Organization an attribute "specificMission: String", which we state will be assigned a value on the level of a particular organization (M_0) only. To support abstraction further still we should allow for level-crossing relations other than the instantiation relation [23]. This means that we are interested in linking objects defined on different classification levels.

Considering the above-mentioned desired features and mechanisms, an application of an alternative language architecture, namely multilevel modeling, seems to be promising. The term multilevel modeling covers any modeling approach that aims to provide systematic support for representing multiple classification levels within a single body of model content [23]. Multilevel modeling is steadily increasing on importance and popularity, cf. [16,24]. Indeed, for different use cases, multilevel modeling leads to a more ‘accurate’ and ‘simpler’ representation of a domain than conventional approaches, cf., [24].

Different multilevel modeling approaches have been proposed, among them, a potency-based multilevel modeling (also called deep instantiation) [22], multilevel object and relations [25], as well as the Flexible Meta-Modeling and Execution Language (FMML^x) [11]. A few features, which we need to model organizational structures, are shared across all approaches: (1) support for arbitrary-depth classification hierarchies, (2) relaxing the type/instance dichotomy, and, (3) offering a deferred instantiation mechanism. However, these approaches differ when it comes to (1) the way these features have been designed, e.g., how the deferred instantiation is defined, and to which modeling elements it may be applied, and (2) additional mechanisms and tool software support, cf. [16]. Considering the need to equip organizational structure models with analysis possibilities, FMML^x, which as the only multilevel modeling approach offers a common representation of model and code and is equipped with a language execution engine, becomes our approach of choice.

FMML^x [11] is based on an extension of XCore [26], which is the meta model of the executable meta modeling facility (XMF) [27]. The flexibility of XMF is, among others, enabled by dynamic typing. XMF supports the specification and implementation of multiple programming and modeling languages. For this purpose, the meta model of a modeling language has to be defined as an instance of XCore. XCore allows for an arbitrary number of classification levels, which is accomplished through a recursive and reflexive language architecture [17]. In XCore every (meta) class is an object and can have a state. FMML^x features a common representation of code and models, i.e., within the FMML^x model elements are classes that allow for the definition of attributes and operations, but at the same time they are also objects. Consequently, we can also assign values and execute operations.

In addition, the definition of attributes, operations and relations has been equipped with *intrinsicness* [11,17] to enable deferred instantiation. Intrinsicness is represented by an integer value that indicates the precise level on which a given attribute, relation or operation will be instantiated, cf. Fig. 2. Finally, it is possible to define various constraints by referring to elements at any abstraction level by using XOCL [26].

FMML^x is supplemented by a meta modeling and programming environment, called XModeler [17], which may be used for creating multilevel models. Next to that XModeler, being an integrated modeling and programming environment, offers the possibility to obtain information from other sources (e.g., to query enterprise data) as well as to execute operations. XModeler offers also additional

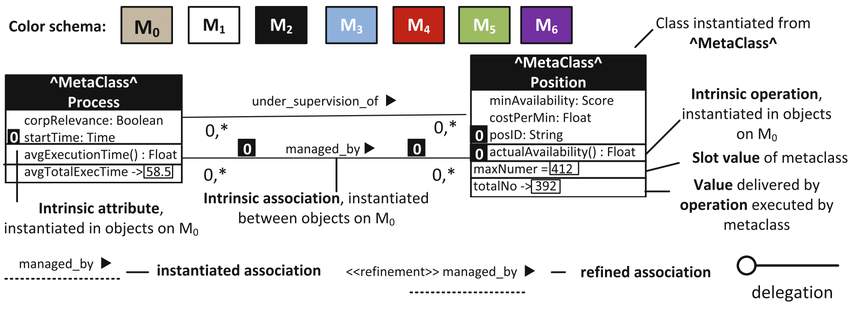

Color schema:
M0
M1
M2
M3
M4
M5
M6
Class instantiated from ^MetaClass^
^MetaClass^ Process
corpRelevance: Boolean
0 startTime: Time
avgExecutionTime() : Float
avgTotalExecTime -> 58.5
under_supervision_of ▶
0,*
0,*
0 managed_by ▶ 0
0,*
0,*
^MetaClass^ Position
minAvailability: Score
costPerMin: Float
0 posID: String
0 actualAvailability() : Float
maxNumer = 412
totalNo -> 392
Intrinsic operation, instantiated in objects on M0
Slot value of metaclass
Value delivered by operation executed by metaclass
Intrinsic attribute, instantiated in objects on M0
Intrinsic association, instantiated between objects on M0
managed_by ▶ — instantiated association
<<refinement>> managed_by ▶ — refined association
delegation

Fig. 2. FMML^x concrete syntax, based on [11]

mechanisms one can benefit from. For instance, it offers the so called 'delegation mechanism' [28]. This means that it is possible to define an association between one object (Role) that provides an access to the state of another object (Role filler) [28].

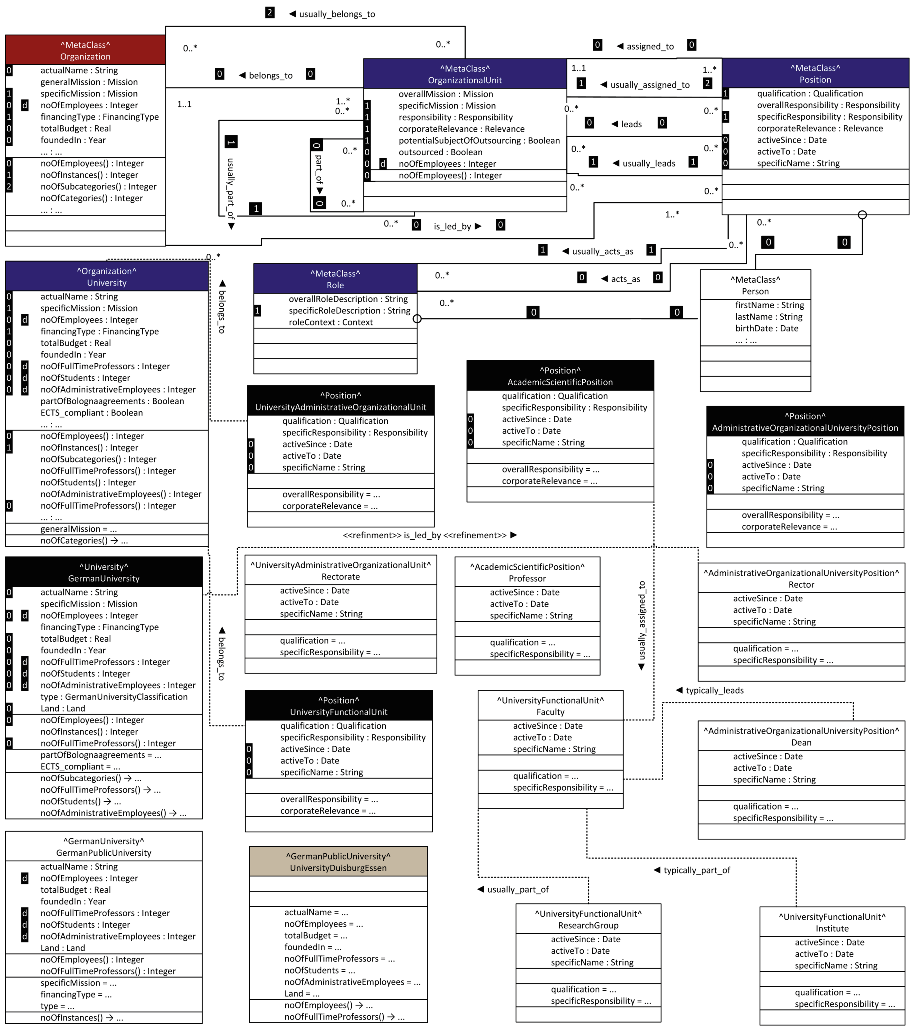
5 Multilevel Model and Resulting Prospects

An excerpt from a multilevel model that represents the university scenario, created using FMML^x, is presented in Fig. 3. Due to space constraints we show only exemplary concepts, exemplary attributes and relationships and do not account for the assigned values of properties.

On the M_4 level, we find the concept *Organization*, whereas on the M_3 level we define such concepts as *OrganizationalUnit*, *Position* and *Role*, which, as discussed in Sect. 3.2, are applicable to any type of organization. Already on this classification level (M_4 or M_3, respectively), we define everything we know about those concepts. For instance, we use intrinsicness to define properties whose values will be known only on lower, not directly proceeding levels. For example, for an *OrganizationalUnit* the attribute "noOfEmployees" has been defined and assigned with the intrinsicness of 0. Hence, its value will be assigned on M_0. Please note as well that for each of those concepts, we define a set of operations to support the organizational structure analyses.

We now instantiate those concepts further to express for organizational structures of universities. While doing so we observe the following advantages of using FMML^x.

Benefiting from the relaxed type/instance dichotomy. While we move along the instantiation hierarchy, we, both (1) specialize the concepts: the structure is inherited and additional properties can be added, and (2) we instantiate the concepts: relevant attributes obtain values and relevant operations can be executed. For example, the class *Organization* is instantiated into *University*. For a *University*, then, we can directly assign a value to the property "generalMission", and add additional properties to its definition (e.g., "numberOfProfessors"). Also, we can use intrinsicness to specify the level at which we instantiate a property.



^MetaClass^ Organization
^MetaClass^ OrganizationalUnit
^MetaClass^ Position
^MetaClass^ Role
^MetaClass^ Person
^Organization^ University
^University^ GermanUniversity
^GermanUniversity^ GermanPublicUniversity
^GermanPublicUniversity^ UniversityDuisburgEssen
^Position^ UniversityAdministrativeOrganizationalUnit
^Position^ AcademicScientificPosition
^Position^ AdministrativeOrganizationalUniversityPosition
^Position^ UniversityFunctionalUnit
^UniversityAdministrativeOrganizationalUnit^ Rectorate
^AcademicScientificPosition^ Professor
^AdministrativeOrganizationalUniversityPosition^ Rector
^AdministrativeOrganizationalUniversityPosition^ Dean
^UniversityFunctionalUnit^ Faculty
^UniversityFunctionalUnit^ ResearchGroup
^UniversityFunctionalUnit^ Institute
usually_belongs_to
belongs_to
assigned_to
usually_assigned_to
leads
usually_leads
usually_part_of
part_of
is_led_by
usually_acts_as
acts_as
<<refinement>> is_led_by <<refinement>>
typically_leads
typically_part_of

Fig. 3. Multilevel model: an excerpt

Moving further along the abstraction hierarchy, once we know that a *University* is a *GermanUniversity*, we can set the attribute *partOfBolognaAgreements* to "true". Finally, for a *ResearchGroup* and *Faculty* (both are instances of *UniversityFunctionalUnit*), the attribute *outsourced* can be set to "false".

Accounting for refinements, to specify allowed relations on a lower level of abstraction. While we move down the hierarchy, using FMML^x we can define refinements (i.e., a type of constraints) of relationship definitions as well as to instantiate them. For instance, a *Position* can lead an *Organization*. However if this *Organization* is a *University*, then usually it is led by a *Rector*. In order to express that a *Rector* (as a particular *Position*) leads a *University* (as a particular

Organization), we define a refinement relationship between these concepts. This allows one to assign only a Rector – so, no other instances of a Position class – to instances of a University class.

Asking analysis questions of the model, based upon current data sets of the running university. Once the model has been implemented using XModeler, thanks to the common representation of model and code, we can keep the model up to date with data of the running university. Thus, we have the possibility to ask analysis questions like, for the University of Duisburg-Essen "what is the number of full time professors?" (through the operation *noOfFulltimeProfessors()* in Fig. 3). Furthermore, given the possibility to express an arbitrary amount of classification levels, we can ask such questions at multiple abstraction levels. For example, using the operation noOfEmployees(), we can request the total number of employees of *UniversityOfDuisburgEssen*, and at the same time, as a (derived) aggregate, we can request noOfEmployees() of all instances of the concept *GermanUniversity.*

Having reuse and productivity at the same time. On the one hand, on M_3 we depict that *Position* may act_as one or more *Role.* This is a typical pattern one finds in organizational modeling (e.g., in ARIS) that is relevant for any type of organization. Among others, for a given position in an organization it can be used to make a fine-grained distinction between the different roles that this position can fulfill, which in turn allows for a differentiated treatment - on the individual role level - of the different responsibilities and qualifications involved in that particular role. For example, we use this abstract pattern to express that *ResearchAssistant* can fulfill, both, the Role of a *Researcher*, and the Role of a *Lecturer*, with both Roles having a differing responsibility. Thus, the information provided on higher classification level can be reused across different types of organizations (model reuse). However, at the same time the additional knowledge added, can be directly used to model domain concepts (in our case the domain of university), which fosters modeling productivity.

6 Conclusions and Research Outlook

In this paper we discussed limitations imposed by the conventional modeling paradigm for modeling organizational structures, particularly concerning compromises with abstraction capabilities, and the syncing of model and code. Subsequently, we showed how a novel modeling paradigm, namely multilevel modeling and integrated modeling and programming, can address these limitations.

Concerning limitations and future work, we compared multilevel modeling to conventional (meta) modeling approaches only, i.e., those approaches that are (roughly speaking) in line with MOF. While this is certainly a useful exercise, we have yet to include alternative modeling approaches that are based on a different language paradigm, such as Object Role Modeling (ORM). ORM also has promising features, like (1) a closeness to natural language, which, next to fostering communication to end users, offers the prospect of mining organizational data and syncing it with the model at hand, (2) ORM offers straightforward parent-child relations as a means of abstraction. Thus, there is no need to account for different "abstraction levels" in addition. As such, a comparison to non-MOF based approaches is warranted – especially for the emerging multilevel modeling community, who tends to differentiate itself mostly from MOF-based thinking. However, even if it sometimes treated as somehow being "fundamental", in the end MOF-based thinking is only one perception of modeling, and a compromised perception at that, as illustrated in this paper.

In addition, we should consider ontologies for organizational structures, cf., [10]. An interesting initiative is a core organizational structure ontology, which recognizes the need to account for multiple classification levels [10]. Here the authors have used the multi-level modeling theory (MLT) [29] together with a foundational ontology (UFO) [30]. This work however differs from ours as the expressiveness of MLT theory differs substantially from the other proposed multilevel modeling approaches, partly in line with the conceptual distinctions made in foundational ontologies (cf. [10]). Therefore, it would be interesting to provide an analysis of where language-based and ontology-based multilevel approaches complement each other.

References

1. Sandkuhl, K., Stirna, J., Persson, A., Wißotzki, M.: Enterprise Modeling: Tackling Business Challenges with the 4EM Method. Springer, Berlin (2014)
2. Frank, U.: Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges. SoSyM **13**(3), 941–962 (2014)
3. Frank, U.: MEMO Organisation Modelling Language (1): Focus on Organisational Structure. ICB-Research Report 48, University of Duisburg-Essen (2011)
4. Daft, R.: Organization Theory and Design. SWC-General Business Series. South-Western College Publishing, New York (2010)
5. The Open Group: ArchiMate 2.1 Specification: Open Group Standard. The Open Group Series. Van Haren, Zaltbommel (2013)
6. Scheer, A.W.: ARIS - Modellierungsmethoden, Metamodelle, Anwendungen, 4th edn. Springer, Heidelberg (2001)
7. SoftwareAG: Aris method manual v. 10 (2017)
8. Sandkuhl, K., Wißotzki, M., Stirna, J.: Unternehmensmodellierung: Grundlagen, Methode und Praktiken. Springer Vieweg, Heidelberg (2013)
9. Frank, U.: Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges. SoSym **13**(3), 941–962 (2012)
10. Carvalho, V.A., Almeida, J.P.A.: A semantic foundation for organizational structures: a multi-level approach. In: 2015 IEEE 19th International Enterprise Distributed Object Computing Conference, pp. 50–59, September 2015)
11. Frank, U.: Multilevel modeling - toward a new paradigm of conceptual modeling and information systems design. BISE **6**(6), 319–337 (2014)
12. Mintzberg, H.: The Structuring of Organizations: A Synthesis of the Research. Theory of Management Policy Series. Prentice-Hall, Englewood Cliffs (1979)
13. Fayol, H.: General and Industrial Management. Pitman Paperbacks, Pitman (1949)
14. van der Linden, D., Hoppenbrouwers, S.: Challenges of identifying communities with shared semantics in enterprise modeling. In: Sandkuhl, K., Seigerroth, U., Stirna, J. (eds.) PoEM 2012. LNBIP, vol. 134, pp. 160–171. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34549-4_12

15. Frank, U.: The MEMO Meta modeling Language (MML) and Language Architecture. 2nd (edn.), ICB-Research Report 43. University of Duisburg-Essen, Essen (2011)
16. Kaczmarek-Heß, M., de Kinderen, S.: A multilevel model of IT platforms for the needs of enterprise IT landscape analyses. BISE (5) (2017)
17. Frank, U.: Designing models and systems to support it management: a case for multilevel modeling. In: Atkinson, C., Grossmann, G., Clark, T., (eds.) MULTI@MoDELS, pp. 3–24 (2016). ceur-ws.org
18. Gustafsson, P., Höök, D., Franke, U., Johnson, P.: Modeling the it impact on organizational structure. In: Proceedings of the 13th IEEE International Conference on Enterprise Distributed Object Computing, EDOC 2009, Piscataway, NJ, USA, pp. 12–21. IEEE Press (2009)
19. Pereira, D., Almeida, J.P.A.: Representing organizational structures in an enterprise architecture language. In: 6th Workshop on Formal Ontologies meet Industry (FOMI 2014), pp. 1–12 (2014)
20. Santos, P.S., Almeida, J.P.A., Guizzardi, G.: An ontology-based analysis and semantics for organizational structure modeling in the ARIS method. Inf. Syst. **38**(5), 690–708 (2013)
21. Frank, U.: Some guidelines for the conception of domain-specific modelling languages. In: Nttgens, M., Thomas, O., Weber, B. (eds.) Proceedings of the Conference 'Enterprise Modelling and Information Systems Architectures' (EMISA 2011). Lecture Notes in Informatics, vol. P-190, Bonn, Germany, GI, pp. 93–106 (2011)
22. Atkinson, C., Kühne, T.: Reducing accidental complexity in domain models. SoSyM **7**(3), 345–359 (2008)
23. Atkinson, C., Kühne, T.: The essence of multilevel metamodeling. In: Gogolla, M., Kobryn, C. (eds.) UML 2001. LNCS, vol. 2185, pp. 19–33. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45441-1_3
24. de Lara, J., Guerra, E., Cuadrado, J.S.: When and how to use multilevel modelling. ACM TOSEM **24**(2), 12:1–12:46 (2014)
25. Neumayr, B., Grün, K., Schrefl, M.: Multi-level domain modeling with m-objects and m-relationships. In: Proceedings of the 6th Asia-Pacific Conference on Conceptual Modeling, Darlinghurst, Australia, Australian Computer Society, pp. 107–116 (2009)
26. Clark, T., Sammut, P., Willans, J.: Applied Metamodelling: A Foundation for Language Driven Development. Ceteva, Sheffield (2008)
27. Clark, T., Willans, J.: Software language engineering with XMF and XModeler. In: Mernik, M. (ed.) Formal and Practical Aspects of Domain-specific Languages: Recent Developments, pp. 311–340. IGI Global (2013)
28. Frank, U.: Delegation: an important concept for the appropriate design of object models. J. Object-Oriented Program. **13**(3), 13–18 (2000)
29. Carvalho, V.A., Almeida, J.P.A.: Toward a well-founded theory for multi-level conceptual modeling. SoSyM **17**(1), 1–27 (2016)
30. Guizzardi, G., Wagner, G.: A unified foundational ontology and some applications of it in business modeling. In: CAiSE 2004 Workshops in Connection with the 16th CaISE Conference, Riga, Latvia, 7–11 June 2004, pp. 129–143. CEUR-WS (2004)

DevOps Competences and Maturity for Software Producing Organizations

Rico de Feijter[1], Sietse Overbeek[1](✉), Rob van Vliet[2], Erik Jagroep[2], and Sjaak Brinkkemper[1]

[1] Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands
{R.deFeijter,S.J.Overbeek,S.Brinkkemper}@uu.nl
[2] Centric, Gouda, The Netherlands
{Rob.van.Vliet,Erik.Jagroep}@centric.eu

Abstract. Software producing organizations aim to release high quality software faster, which triggers the adoption of DevOps. However, not many artifacts are available that aid in adopting DevOps. In an attempt to bridge this gap, a DevOps Competence Model showing an overview of the areas to be considered in adopting DevOps is proposed. Also, a DevOps Maturity Model is proposed that presents a growth path for software producing organizations. Both these models incorporate perspectives that are made up of focus areas which in turn are made up of capabilities. Apart from designing and validating these models by means of expert workshops, a case study has been conducted where assessees answered questions to gain insight into which capabilities were implemented. From the answers, maturity profiles were extracted that supported the assessees in becoming more DevOps mature.

Keywords: Competence model · Design science · DevOps Maturity model · Software producing organizations

1 Introduction

Software producing organizations (SPOs) are moving away from on-premise software to cloud-based software that allows for faster releasing [13,16]. However, striving for faster releasing against a high quality means that stakeholders in SPOs should collaborate more closely and in order to achieve this DevOps provides a helping hand [23]. The term stresses improving collaboration between stakeholders such as development (Dev), operations (Ops), product management, and quality assurance [11] with the aim to provide the customer with high quality releases by embracing practices related to creating a healthy culture and improved collaboration. Moreover, the term addresses automating tasks, lean thinking, and continuous improvement by leveraging monitoring and measurement [14]. Despite the increasing popularity of DevOps and organizations having an understanding of the motivations to adopt DevOps and the advantages the

J. Gulden et al. (Eds.): BPMDS 2018/EMMSAD 2018, LNBIP 318, pp. 244–259, 2018.
https://doi.org/10.1007/978-3-319-91704-7_16

notion brings [18], DevOps requires further investigation as there is no clear overview of DevOps practices. This causes organizations to discover for themselves how to adopt DevOps [4].

The scientific contributions of this research are as fourfold. First, a set of drivers and capabilities is proposed to establish a set of practices and to provide support for DevOps adoption. Second, based on these drivers, capabilities, and practices a DevOps competence model has been designed as is shown in Sect. 3. This DevOps competence model shows which focus areas and perspectives are of importance when adopting DevOps. Third, a DevOps maturity model has been designed which is presented in Sect. 4, which is used to indicate on what level of DevOps maturity a SPO can be positioned and what needs to be done in order to reach higher levels of DevOps maturity. Fourth, and lastly, a case study has been conducted in which different business units as part of a SPO have performed a self-assessment to determine their DevOps maturity as is shown in Sect. 5. For this purpose, we have developed a DevOps assessment tool that has been used to perform these self-assessments. The paper ends with conclusions and future research in Sect. 7. Before explaining the results of this research, the research approach is discussed next.

2 Research Approach

This study was executed at Centric, a large SPO located in the Netherlands and followed a design science approach [8]. A brief overview of how design science was applied to conduct this research is provided in Fig. 1, which shows the steps involved in the research and references to the paragraphs that explain the research approach more in detail.

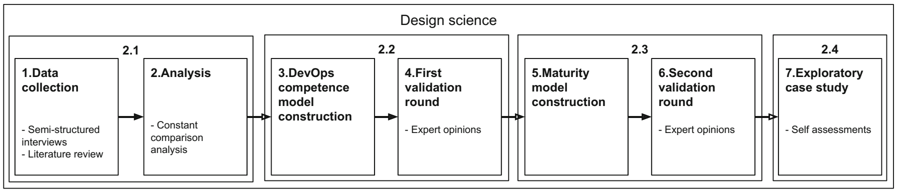


Fig. 1. Research approach

2.1 Data Collection and Analysis

In the data collection phase, a literature review, and semi-structured interviews were conducted to identify DevOps drivers providing an understanding of the motivations to adopt DevOps. The resulting drivers concerned Collaboration Culture, Agility and Process Alignment, Automation, Quality, Development and Deployment of Cloud Based Applications, and Continuous Improvement. Aside from drivers, capabilities that support maturing in DevOps are identified.

A full list of all identified capabilities is shown in the appendix. Keywords that were used to obtain drivers from literature concerned DevOps, DevOps drivers, DevOps motivation, need for DevOps, 'why' DevOps, and DevOps objectives. The drivers found in literature have been used to form an interview protocol in which the identified drivers acted as a guide to elicit capabilities from practice. In total, 14 interviews were held at 3 SPOs and parts of the transcripts are validated by the interviewees through e-mail and follow-up interviews. Parallel to these interviews, a literature review was done to find DevOps capabilities by using keywords that adhered to the following structure: DevOps [keyword] practices OR patterns OR principles, where [keyword] was replaced by the words lean, continuous improvement, automation, quality, culture, collaboration, and alignment. DevOps practices are in some cases also known as DevOps principles and patterns in literature implying that patterns and principles have also been adopted in the search string. After obtaining data by conducting a literature review and semi-structured interviews, the data were analyzed by means of constant comparison analysis, which enables the identification of themes in qualitative data [15,17]. This technique helped to realize the drivers and the initial capabilities, which are abstracted to focus areas that are defined in [19] as defined subsets of a functional domain which is DevOps in this case. Eventually, focus areas are abstracted to perspectives. Important to note is that the perspectives, focus areas, and capabilities not only emerged from the literature review results and interviews, as the validation rounds and the case study also contributed to their existence.

2.2 DevOps Competence Model Construction and Validation

Subsequently, a DevOps competence model showing the areas to focus on to adopt DevOps was constructed on the basis of the earlier obtained capabilities, focus areas, perspectives, literature and inherently also the drivers. To ensure credibility of the DevOps competence model and the perspectives, focus areas, and capabilities, a first validation round was executed. This first validation round covered expert opinions [25] in the form of a workshop and four follow-up validation sessions. First, a workshop was held with seven DevOps experts. During this workshop session, the DevOps competence model was explained after which it was validated against criteria, which encompassed understandability and clarity of the model. Thereafter, the perspectives, focus areas, and capabilities were validated against understandability, relevance, and completeness of each focus area and its corresponding capabilities. The correctness of the maturity order of each set of capabilities belonging to a focus area was also discussed. After the workshop, four follow-up validation sessions were executed with two of the participants from the workshop session in order to validate the processed input from the workshop. The final version of the DevOps competence model is shown and described in Sect. 3.

2.3 Maturity Model Construction and Validation

Next, the validated perspectives, focus areas, capabilities, assessment data, literature, and interview data served as input for the DevOps maturity model, which is a DevOps maturity measurement instrument and supports maturing in DevOps in a step-by-step way. After creating the DevOps maturity model by transferring the validated perspectives and focus areas to the DevOps maturity model and positioning the capabilities in the DevOps maturity model, a second validation round involving expert opinions took place. This validation round covered a validation of the DevOps maturity model together with a second validation of the perspectives, focus areas, and capabilities. Four experts validated the correctness of the positioning of the capabilities in the DevOps maturity model by asking whether the expert agreed with the positioning of the capabilities within a focus area (i.e. intra-dependencies [21]). Further, the DevOps maturity model was also validated by one of the authors of [21] at a later stage. This session brought forward the acknowledgement of dependencies among capabilities from different focus areas, i.e., interdependencies. In following this input, interdependencies were added to the DevOps maturity model to the extent these could be detected. The final version of the DevOps maturity model is shown and described in Sect. 4.

2.4 Exploratory Case Study

The last part of the study encompassed the execution of a multiple holistic exploratory case study by adhering to a protocol as advocated in [26]. The case study served as a means to justify the application of the capabilities and the constructed DevOps maturity model. Moreover, the exploratory case study comprised forty-five questions that were based on the capabilities. These questions were part of a self assessment that was sent out to nineteen assessees from the organization Centric at which this study took place. These assessees belonged to highly divergent business units and each assessee was involved in the creation of a certain product. In total, eight assessees filled out the self assessment and seven of the eight self assessments were found useful and were used to make up maturity plots [19]. The case study also gave rise to an extra capability (see Infrastructure - Capability B in the appendix) and is further detailed in Sect. 5.

3 A DevOps Competence Model

A DevOps competence model was constructed, which is shown in Fig. 2. The model represents a software house, which is an organization involved in product development [3] and comprises three overarching perspectives that cover focus areas. The perspectives, which are Culture and Collaboration (CC), Product, Process and Quality (PPQ), and Foundation (F) are discussed below.

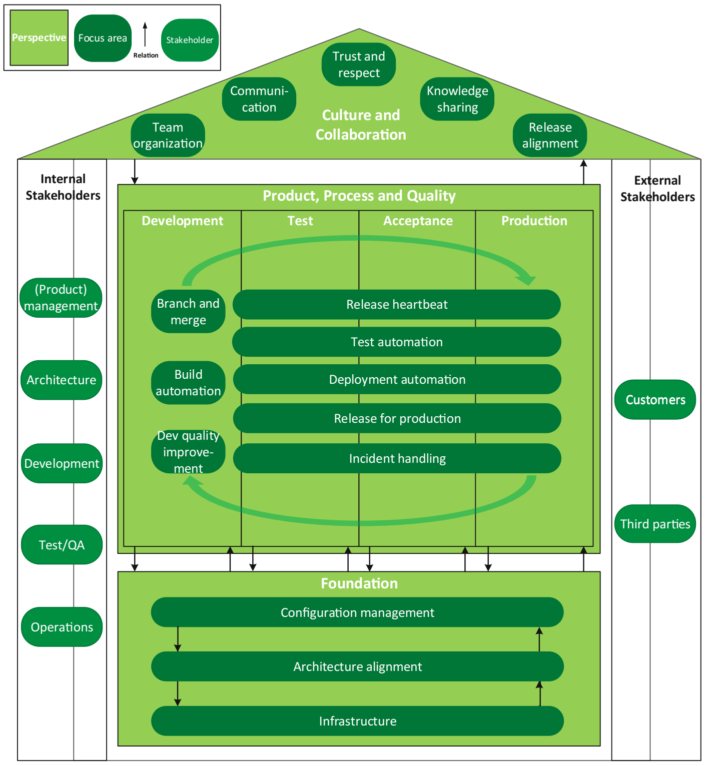


Fig. 2. The DevOps competence model (Color figure online)

3.1 Culture and Collaboration

The Culture and Collaboration perspective covers the soft part of DevOps and forms the 'roof of the house'. Moreover, the perspective covers five focus areas, which are: Team Organization, Communication, Trust and Respect, Knowledge Sharing, and Release Alignment. This perspective reflects a prominent part of the model, because the organization itself should be in place to perform work. In order to perform work, interdisciplinary professionals should at least communicate indirectly with each other. Ideally, professionals also share knowledge, have trust and respect for one another, work in teams, and there should be some form of alignment between internal and external dependencies in order to timely deploy software.

3.2 Product, Process and Quality

The PPQ perspective visualizes the process of releasing a product and feedback loops. Moreover, when viewing this perspective, it can be discerned that the focus area in this layer touches upon four environments that together represent the DTAP-street, which is an acronym for development, testing, acceptance, and production and represents the environments on which development, testing, and running software in production occurs. The DTAP-street is a well-known industry practice [7], which makes its inclusion in the DevOps competence model understandable and recognizable for practitioners.

The model is thus set up in such a way that the focus areas branch and merge, build automation, and development quality improvement are mainly concerned with development and thus reside in the development environment. However, the release heartbeat, test automation, deployment automation, release for production, and incident handling focus areas are concerned with the remainder of the environments as well. For instance, when looking at release heartbeat, requirements can be gathered from production and validation of functionality can occur on testing and acceptance environments but also on production environments. Testing, on the other hand, could involve regression or unit tests, acceptance tests, or resilience testing which occurs in production. Deployment automation touches upon all environments and release for production concerns all environments as software is only declared done in a DevOps context once it is developed according to wishes, passed all tests, works in production, and leverages value to the customer. Lastly, incident handling involves repairing an incident, testing the fix, and releasing the fix. Meanwhile, production needs to be monitored to pro-actively act on incidents.

When further scrutinizing the environments, a green arrow can be perceived that explicitly denotes a feedback loop. This arrow explicates that software is continuously pushed to production, while at the same time usage data from production is fed back to product management in order to better comply with customers wishes. Note, however, that the arrow not only concentrates on the feedback loop from production to product management, since along the way to production, other feedback loops can be observed as well. For instance, when a test on the testing environment or the acceptance environment fails, developers might have to fix the code after which the tester needs to perform another test on the software build. Furthermore, relations show that the Product, Process and Quality perspective relates to the Culture and Collaboration perspective, while this also applies the other way around. Indeed, when an incident comes in, product management, developers, testers, and operations should communicate such that the resulting fix can be put into production again in a timely fashion.

3.3 Foundation

The foundation perspective encompasses the configuration management, architecture and infrastructure focus areas that stretch from development to production and aim to support the process depicted in the product, process and quality perspective. The reason for stretching these in such a way is underpinned by the fact that for all displayed environments configuration items such as OS, middleware, database versions and so on should be managed [9]. Additionally, each environment has a technical architecture, which is also concerned with the software architecture. Also, infrastructure inherently mirrors all environments, as environments are a representation of infrastructure [9]. The relation between the foundation perspective and four environments is also clarified by the arrows between the Product, Process and Quality, and Foundation perspectives.

Next to the aforementioned, the three focus areas at the bottom are associated with one another. That is, provisioning infrastructure with the correct configuration items to make environment ready for deployment requires configuration management. Indeed, configurations with which environments are provisioned are retrieved from a certain location, be it an Excel document or a version control system in which the configuration items are managed. Further, architecture alignment is associated with configuration management, since the architecture, i.e., both software and technical, describes the configuration including source code and infrastructure configurations such as middleware configurations at a higher abstraction level [6,22]. Additionally, the relation between configuration management and architecture is made clear as adaption of the configuration leads to adaption of the architecture and vice versa. Besides, when looking at the relation between the infrastructure and the architecture, an architecture describes the structure of the infrastructure at a higher abstraction level [6,12]. The adaption of either of these leads to a modification of the other.

As can be inferred from the aforementioned description the earlier mentioned drivers are reflected in the model and, apart from that, the DevOps competence model incorporates internal and external stakeholders that reside in a DevOps context. Internal stakeholders are represented by management, such as unit managers and product management, who look after requirements management and release planning related activities, among others. Further, software and technical architects are represented by architecture and are concerned with the structure of the software and the technical landscape on which the software should land, respectively. Further, the model includes developers, testers (including information security), and Ops professionals. External stakeholders are customers and third parties from which software is used in the development of a product.

The DevOps competence model as depicted in Fig. 2 attempts to capture the different focus areas DevOps touches upon by presenting these in the form of perspectives. The model also attempts to illustrate the coherency of the perspectives and focus areas, which are cultural, procedural, and technical in nature. The model also makes clear which stakeholders could be involved in DevOps.

4 DevOps Maturity Model

The DevOps maturity model as is shown in Fig. 3 includes focus areas and enables a SPO to mature in a fine grained way, which is in contrast with the CMM model that enables organizations to mature in a generic way [19]. The focus area maturity model also allows for more than five maturity levels to be distinguished and dependencies to exist among capabilities. When observing Fig. 3 more closely, the relation with the DevOps competence model becomes directly clear as the perspectives and focus areas are also present in this model.

The letters residing in the DevOps maturity model represent the sixty-three capabilities detected throughout this study. Note, however, that excerpts of these capabilities can be found in the appendix, while full descriptions of the capabilities can be consulted in [5]. In the maturity model these capabilities are positioned in increasing order of maturity. For example, Build automation - B is more mature than Build automation - A, which makes that Build automation - B is positioned further to the right.

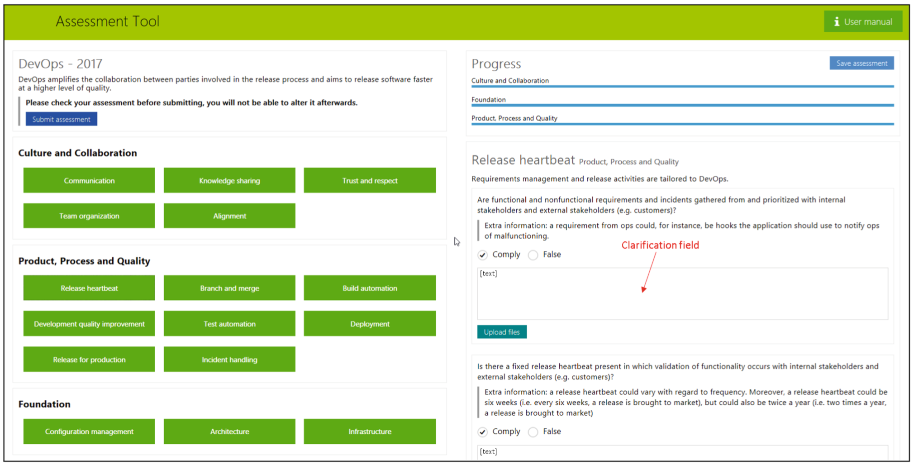
Also, ten levels were determined together with the positioning of the capabilities by taking into account situations that were observed, while conducting interviews and scrutinizing earlier assessment data reflecting the interview situations. Moreover, premature observed situations that were not yet adopting DevOps gave rise to the positioning of the capabilities in levels 1 to 5. A situation transitioning to a DevOps situation gave input to the positioning of the capabilities in levels 5 and 6. Situations in which DevOps already was adopted to a more mature extent gave input to the positioning of the capabilities in levels 5 to 10. Further, positioning was also determined by literature and by taking into account dependencies among capabilities. For instance, it makes sense to first gather and prioritize requirements (Release heartbeat - A) before creating a software build (Build automation - A). Hence, Release heartbeat - A is placed one level lower than Build automation - A.

Focus area \ Level	0	1	2	3	4	5	6	7	8	9	10
Culture and collaboration											
Communication		A				B	C			D	E
Knowledge sharing				A		B	C				D
Trust and respect							A	B	C		
Team organization		A	B						C	D	
Release alignment				A					B	C	
Product, Process and Quality											
Release heartbeat		A				B	C		D	E	F
Branch and merge			A	B		C		D			
Build automation			A	B		C					
Development quality improvement			A			B		C	D	E	
Test automation				A	B	C			D		E
Deployment automation					A	B		C			D
Release for production					A			B	C	D	
Incident handling			A					B	C	D	
Foundation											
Configuration management			A	B		C					
Architecture alignment			A					B			
Infrastructure				A			B	C	D		

Fig. 3. The DevOps maturity model

5 Measuring of DevOps Maturity

A case study was carried out at Centric to justify the capabilities and DevOps maturity model in practice. As said in Sect. 2, the case study yielded seven useful filled in assessments, which were transformed into maturity plots. This transformation was done by qualitatively analyzing the responses from the assessees, who were able to answer if a capability was implemented or not and give an extra explanation to clarify their answer by using the DevOps assessment tool that is shown in Fig. 4. The answers in conjunction with their explanation thus made clear what capabilities were implemented, which made it possible to create


Assessment Tool

User manual

DevOps - 2017

DevOps amplifies the collaboration between parties involved in the release process and aims to release software faster at a higher level of quality.

Please check your assessment before submitting, you will not be able to alter it afterwards.

Submit assessment

Culture and Collaboration

Communication | Knowledge sharing | Trust and respect | Team organization | Alignment

Product, Process and Quality

Release heartbeat | Branch and merge | Build automation | Development quality improvement | Test automation | Deployment | Release for production | Incident handling

Foundation

Configuration management | Architecture | Infrastructure

Progress

Save assessment

Culture and Collaboration

Foundation

Product, Process and Quality

Release heartbeat Product, Process and Quality

Requirements management and release activities are tailored to DevOps.

Are functional and nonfunctional requirements and incidents gathered from and prioritized with internal stakeholders and external stakeholders (e.g. customers)?

Extra information: a requirement from ops could, for instance, be hooks the application should use to notify ops of malfunctioning.

[✓ Comply] False

Clarification field

[text]

Upload files

Is there a fixed release heartbeat present in which validation of functionality occurs with internal stakeholders and external stakeholders (e.g. customers)?

Extra information: a release heartbeat could vary with regard to frequency. Moreover, a release heartbeat could be six weeks (i.e. every six weeks, a release is brought to market), but could also be twice a year (i.e. two times a year, a release is brought to market)

[✓ Comply] False

[text]

Fig. 4. DevOps assessment tool

maturity plots. Two of the cases for which maturity plots were made are adopted in this paper and are shown in Figs. 5 and 6. The choice for adopting these cases lies in the fact that no space is available to adopt all seven cases and that the first case shows a less mature detected situation, whereas the second case shows the most mature situation as found in the context of the case study. Both maturity plots show a green bar that demonstrates the extent to which capabilities have been implemented and also displays the next steps to take in order to grow more mature. A concrete description for each of these cases follows in which no contextual information is provided due to confidentiality reasons (Table 1).

Table 1. Case information

	Product	Age	Number of teams
Case1	On-premise product	23	2
Case2	Cloud product	3	8

The case in which professionals were working on an on-premise product is shown in Fig. 5. This product was hosted at the customer's site and was twenty-three years of age. In total, two teams worked on the product. When looking at the corresponding plot of the case, the observation is made that this case reached a maturity level of two, since all level 2 capabilities were adopted. In order to grow towards level 3, dependencies with teams that deliver shared components (i.e., components that are developed by other teams and used in multiple products) are advised to be taken into account in the road map (Release alignment - Capability A) and configuration items are advised to be managed in tooling rather than in documents (Configuration management - Capability B).

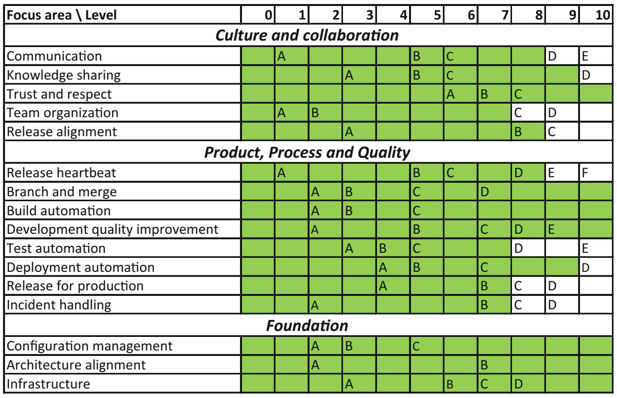
Focus area \ Level	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Culture and collaboration											
Communication	🟩	🟩 A	🟩	🟩	🟩	B	C			D	E
Knowledge sharing	🟩	🟩	🟩	🟩 A		B	C				D
Trust and respect	🟩	🟩	🟩	🟩	🟩	🟩	A	B	C		
Team organization	🟩	🟩 A	🟩 B	🟩	🟩	🟩	🟩	🟩	C	D	
Release alignment	🟩	🟩	🟩	A					B	C	
Product, Process and Quality											
Release heartbeat	🟩	🟩 A	🟩	🟩	🟩	🟩 B	C		D	E	F
Branch and merge	🟩	🟩	🟩 A	🟩 B	🟩	🟩 C	🟩	🟩 D	🟩	🟩	🟩
Build automation	🟩	🟩	🟩 A	🟩 B	🟩	🟩 C	🟩	🟩	🟩	🟩	🟩
Development quality improvement	🟩	🟩	🟩 A	🟩	🟩	🟩 B	🟩	🟩 C	D	E	
Test automation	🟩	🟩	🟩	🟩 A	🟩 B	C			D		E
Deployment automation	🟩	🟩	🟩	🟩	🟩 A	🟩 B	🟩	C			D
Release for production	🟩	🟩	🟩	🟩	🟩 A	🟩	🟩	B	C	D	
Incident handling	🟩	🟩	🟩 A	🟩	🟩	🟩	🟩	B	C	D	
Foundation											
Configuration management	🟩	🟩	🟩 A	B		C					
Architecture alignment	🟩	🟩	🟩 A	🟩	🟩	🟩	🟩	B			
Infrastructure	🟩	🟩	🟩	🟩 A	🟩	🟩	B	C	D		

Fig. 5. Case 1 DevOps maturity profile (Color figure online)

Figure 6 shows a more mature case. Here, professionals were working on a cloud solution of three years of age. This product was either hosted at an own data center or at the customers premises and eight teams worked on this product. As can be inferred from the maturity profile, this case already reached level 7 on the DevOps maturity model and even shows that a number of level 8 capabilities were implemented. However, although many tasks were carried out in a cross functional manner including Ops (e.g., setting up system concepts and load tests), no cross functional teams with Ops were present. An advise pertaining to this case is thus to involve Ops in the cross functional teams (Team organization - Capability C) that already consisted of developers and testers. Other advises to this case in order to grow more mature involve adopting automated acceptance testing and conduct these systematically (Test automation - Capability D) and adopting a definition of done that stretches to the customer, so that a feature is only declared done once it yields customer value (Release for production - Capability C). Finally, blameless postmortems (Incident handling - Capability C) is advised to be implemented to completely attain level 8.

6 Discussion and Limitations

As has been mentioned in the introduction of this paper, no processes and methods were observed that concentrate on the adoption of DevOps. An aim of this work was thus to fill this gap. To this end, a DevOps competence model showing the focus areas to be considered in order to implement DevOps from a balanced perspective and a DevOps maturity model showing a fine grained approach to grow towards a more mature DevOps situation were created. It is safe to say that these artifacts filled the identified gap to a slight extent, since the artifacts



Focus area \ Level	0	1	2	3	4	5	6	7	8	9	10
Culture and collaboration											
Communication	🟩	🟩 A	🟩	🟩	🟩	🟩 B	🟩 C	🟩	🟩	D	E
Knowledge sharing	🟩	🟩	🟩	🟩 A	🟩	🟩 B	🟩 C	🟩	🟩	🟩	D
Trust and respect	🟩	🟩	🟩	🟩	🟩	🟩	🟩 A	🟩 B	🟩 C	🟩	🟩
Team organization	🟩	🟩 A	🟩 B	🟩	🟩	🟩	🟩	🟩	C	D	
Release alignment	🟩	🟩	🟩	🟩 A	🟩	🟩	🟩	🟩	🟩 B	C	
Product, Process and Quality											
Release heartbeat	🟩	🟩 A	🟩	🟩	🟩	🟩 B	🟩 C	🟩	🟩 D	E	F
Branch and merge	🟩	🟩	🟩 A	🟩 B	🟩	🟩 C	🟩	🟩 D	🟩	🟩	🟩
Build automation	🟩	🟩	🟩 A	🟩 B	🟩	🟩 C	🟩	🟩	🟩	🟩	🟩
Development quality improvement	🟩	🟩	🟩 A	🟩	🟩	🟩 B	🟩	🟩 C	🟩 D	🟩 E	🟩
Test automation	🟩	🟩	🟩	🟩 A	🟩 B	🟩 C	🟩	🟩	D		E
Deployment automation	🟩	🟩	🟩	🟩	🟩 A	🟩 B	🟩	🟩 C	🟩	🟩	D
Release for production	🟩	🟩	🟩	🟩	🟩 A	🟩	🟩	🟩 B	C	D	
Incident handling	🟩	🟩	🟩 A	🟩	🟩	🟩	🟩	🟩 B	C	D	
Foundation											
Configuration management	🟩	🟩	🟩 A	🟩 B	🟩	🟩 C	🟩	🟩	🟩	🟩	🟩
Architecture alignment	🟩	🟩	🟩 A	🟩	🟩	🟩	🟩	🟩 B	🟩	🟩	🟩
Infrastructure	🟩	🟩	🟩	🟩 A	🟩	🟩	🟩 B	🟩 C	🟩 D	🟩	🟩

Fig. 6. Case 2 DevOps maturity profile (Color figure online)

came not without limitations. The first limitation concerns the fact that twelve of the interviews were conducted at the same organization, which causes interview data to be biased towards one organization. Also, all participants involved in the first validation round came from the same organization. This limitation impacted the generalizability of the results. Another limitation is that none of the participants participating in the first round validation was a real DevOps expert, although all participants had affinity with sub domains of DevOps.

A limitation in the second validation round is that validation sessions with the interviewees and experts were conducted separately. This caused criteria to be made up to process validation input, as no consensus reaching among participants could be reached. When moving on to the validation of the DevOps maturity model, the validation of the positioning of the capabilities in the DevOps maturity model was done in parallel with the validation of the capabilities themselves in a second validation round due to time constraints. However, validating both the capabilities and the positioning of the capabilities in parallel formed a limitation within this research. Indeed, additions to existing capabilities solely recognized by interviewees could not be validated properly in conjunction with the capabilities their positions by the experts, as the contents of the capabilities were not updated, while executing the second validation round. The capabilities for which this was the case were Development quality improvement - Capability A and Infrastructure - Capability C. Additionally, capability B, C, and D from Release for production and their positions were not validated, since these capabilities arose as new capabilities during the second validation round. Also, the case study impacted the capabilities and their positioning, since Infrastructure - Capability B was added and positioned and Configuration management - Capability C was repositioned due to newly gained insights from the conducted case study. However, this also yielded the fact that Infrastructure - Capability B, its position and the newly assigned position from Configuration management - Capability C were not validated, as the case study occurred after the second validation round. Next, albeit dependencies between capabilities from different focus areas (i.e. interdependencies) were advocated to be adopted in the DevOps maturity model during a validation session, the identified interdependencies themselves have not been validated and also affected dependencies between capabilities residing in the same focus area (i.e. intradependencies) that were validated earlier with the 4 experts.

Lastly, a constraint of the case study is that the questions were scoped to Dev and Ops, thereby leaving out other interdisciplinary professionals. Also, a number of capabilities that were assumed not to be present in the case organization were left out to create more to the point questions and yield a greater chance of obtaining higher response. When further observing the case study limitations, a key remark is that only 8 filled in assessments were obtained. This could be considered a limitation, as more response might have better evaluated the maturity ordering of the capabilities, as in one case Development quality improvement - capability D was implemented, while Development quality improvement - capability C was not. This might imply an incorrect maturity order. Aside from the aforementioned, the case study was conducted in different contexts but in one organization, which poses a limitation on the generalizability of the case study results.

7 Conclusions and Future Research

To become more DevOps mature, a SPO can utilize the DevOps competence model to gain an understanding of the areas to be considered when adopting DevOps and leverage the DevOps maturity model to measure its current DevOps maturity level and determine the next steps to become more DevOps mature. Hence, the combination of the set of capabilities, the DevOps competence model and the DevOps maturity model can be considered the key contributions of this study that contribute to bridging the gap mentioned in [4], where it is concretely stated that there is no clear overview of DevOps practices indicating that organizations are left to discover for themselves how to adopt DevOps. Further, Sect. 6 showed that the research came with a number of limitations that form input to future research. First, to ensure a richer body of drivers and capabilities requires the execution of interviews at multiple independent SPOs. Second, to generalize the DevOps competence model, the model needs to be validated by experts, who are known for their DevOps expertise and come from different SPOs. Third, an amount of capabilities and their corresponding positions require better validation. Fourth, further validation of the intra and interdependencies could be part of further future work. These validations should preferably occur in a group setting instead of an individual setting to be able to reach group consensus. Fifth and last, all capabilities should be evaluated in a case study to their fullest extent and to better evaluate maturity ordering of the capabilities, a broader case study should be done. Finally, future work could aim at studying the intertwinement of high level product management processes such as portfolio management with DevOps and situational factors, which consider the context of a SPO and are of use to determine the correct set of capabilities to be implemented in a certain SPO context [1].

Appendix: Focus areas and Capabilities

CC - Communication

A. Indirect communication communication between interdisciplinary professionals, among which are Dev and Ops professionals, is indirectly established (e.g. through managers, procedures). **B. Facilitated communication** direct communication between interdisciplinary professionals, among which are Dev and Ops professionals, is facilitated by management by stimulating professionals to communicate directly. **C. Direct communication** direct interdisciplinary communication between professionals, among which are Dev and Ops professionals while working towards a release is present. This direct communication could occur through mailing lists, personal contact etc. **D. Structured communication** a structure for interdisciplinary communication is in place (e.g. by holding daily standups and retrospectives with interdisciplinary professionals including Dev and Ops, and by maintaining contact with (product) management to discuss about impediments along the way, work to be done the upcoming sprints, and the technical debt situation, among others). **E. Communication improvement** communication among management and interdisciplinary professionals, including Dev and Ops, is improved (e.g. by adopting and trying out new communication practices from industry, learning from experiences and by tracking projects or using instruments such as skill matrices and peer feedback mechanisms over time).

CC - Knowledge sharing

A. Decentralized knowledge sharing knowledge is shared between interdisciplinary professionals, among which are Dev and Ops professionals in a decentralized way (i.e. through notes or documents). **B. Centralized knowledge sharing** knowledge is shared between interdisciplinary professionals, among which are Dev and Ops professionals, through centralized knowledge sharing facilities. **C. Active knowledge sharing** knowledge is shared actively between interdisciplinary professionals, among which are Dev and Ops professionals.
D. Communities of practice knowledge is shared through communities of practice, which are composed of multidisciplinary professionals that share a common interest.

CC - Trust and respect

A. Culture of trust and respect imitation dynamics, level of autonomy, and planning are open for collaboration and creation of trust and respect between interdisciplinary professionals, among which are Dev and Ops people. An example here is a DevOps duty rotation where developers take on operational tasks. **B. Culture of trust and respect facilitation** a culture of trust and respect is facilitated by management. Facilitation by management means that management should not manage by fear, but should act as a servant leader that supports professionals in day-to-day tasks, has an understanding of operational tasks, and allows interdisciplinary professionals, among which are Dev and Ops professionals, to learn quickly from mistakes. **C. Culture of trust and respect shared core values** the culture of trust and respect between interdisciplinary professionals, among which are Dev and Ops professionals, is maintained by following shared core values such as rewarding Dev and Ops as a group when a release is successful, being transparent and open towards one another to prevent blaming, and working towards shared goals.

CC - Team organization

A. Separate teams separate teams are present (e.g. development teams, operations teams etc). **B. Cross functional teams excluding Ops** cross functional teams are present that exclude operations (e.g. teams consisting of developers and testers are present). **C. Cross functional teams including Ops** cross functional teams are present that include operations. **D. Cross functional teams with knowledge overlap** cross functional teams are present in which professionals have boundary crossing knowledge (e.g. T-shaped professionals that have Dev and Ops knowledge).

CC - Release alignment

A. Roadmap alignment alignment with dependent internal and external stakeholders (e.g. third parties) is considered in the roadmap. **B. Internal release heartbeat alignment** the release heartbeat is aligned with dependent internal stakeholders. An example of such an alignment could be reflected in adopting the same deployment moments or adhering to a common sprint cadence. **C. External release heartbeat alignment** the release heartbeat is aligned with dependent external stakeholders such as third parties from which software is used in the development of a product.

PPQ - Release heartbeat

A. Requirements and incidents gathering and prioritization Functional and nonfunctional requirements and incidents are gathered from and prioritized with internal stakeholders and external stakeholders (e.g. customers). **B. Fixed release heartbeat and validation** a fixed release heartbeat is present and validation of functionality occurs with internal stakeholders and external stakeholders (e.g. customers) by demoing the functionality on a test or acceptance environment or the like. **C. Production requirements and incident gathering** functional and nonfunctional requirements and incidents are gathered from production by monitoring the production environment(s). **D. Gradual release and production validation** functionality is released gradually (e.g. functionality is first released to internal stakeholders, whereafter it is released to stakeholders that have close bonds with the organization. Finally, the software is released to end-customers) and validation of functionality occurs in production. **E. Feature experiments** experiments are run with slices of features in order to support the prioritization of the contents in the backlog (e.g. A/B testing). **F. Release heartbeat improvement** the value stream is continuously improved by identifying and eliminating activities that do not add any value, shortening lead times and shortening feedback loops such as the time between feedback moments with the customer.

PPQ - Branch and merge

A. Version controlled source code source code is stored under version control. **B. Branching/merging strategy** a branching/merging strategy is adhered to that allows multiple developers to collaborate and allows code to be branched and merged. **C. DevOps branching/merging strategy** a branching/merging strategy is adhered to that is DevOps compatible. An example of such a strategy is trunk based development. **D. Feature toggles** feature toggles are used to release functionality to customers by making completed functionality available.

PPQ - Build automation

Manual build creation a software build is created manually. **Automated build creation** a build is created automatically (e.g. by running a scheduled build at night). **Continuous build creation** a CI build is created after each check-in to verify that the integrated code still yields a working software build.

PPQ - Development quality improvement

A. Manual code quality monitoring manual code quality improvement mechanisms are in place such as pair programming, code reviews, and adherence to code conventions. **B. Broken build detection** broken software builds are detected, made visual and quickly repaired. **C. Gated check-in** gated check-ins are performed. **D. Automated code quality monitoring** code quality is monitored automatically (e.g. automated code reviews). **E. Quality gates** quality gates are defined against which the quality of code is measured.

PPQ - Test automation

A. Systematic testing Manual unit and acceptance tests are performed systematically. **B. Advanced systematic testing** manual integration (chain) and regression tests are performed systematically and test driven development practices are used in testing such as using mocking frameworks and writing unit tests before writing code. **C. Automated systematic testing** automated unit and nonfunctional tests are performed systematically. **D. Advanced automated systematic testing** automated regression, integration (chain) and acceptance tests are performed systematically. **E. Automated recoverability and resilience testing** automated recoverability and resilience tests are randomly performed in production.

PPQ - Deployment automation

A. Manual deployment software is deployed to environments in a manual fashion. In addition, rollback is possible, where data is brought back to a stable state. **B. Partly automated deployment** software is deployed automatically to some environments. **C. Continuous delivery** deployment to all environments occurs in an automated manner (e.g. via self service deployments), where data model changes are also processed automatically. **D. Continuous deployment** each check-in is continuously deployed to production, where data model changes are also processed and automated rollback is possible.

PPQ - Release for production

A. Definition of done a definition of done that incorporates development and testing criteria, among others to be complied with during a sprint, is followed. **B. Definition of release** a definition of release that incorporates Ops criteria (e.g. verifying whether the software works in production) to be complied with before releasing to customers, is followed. **C. Done according to customer** functionality is declared done when customer satisfaction has been reached. **D. Automated material generation** Supporting materials such as release documentation, training documentation etc. are automatically generated.

PPQ - Incident handling

A. Reactive incident handling incidents are reactively acted upon by interdisciplinary professionals, among which are Dev and Ops professionals. **B. Proactive incident handling** incidents are proactively acted upon by interdisciplinary professionals, among which are Dev and Ops professionals **C. Blameless root cause detection** root causes are identified without blaming one another by conducting blameless postmortems involving both Dev and Ops. **D. Automated root cause detection** the identification of root causes of incidents is supported by analytics.

F - Configuration management

A. Manual configuration management Supported versions of configuration items (e.g. OS, middleware etc.) and their relationships are managed manually, for instance in documents or excel sheets. **B. Automated configuration management** Supported versions of configuration times and their relationships are managed in a configuration management tool. **C. Version controlled configuration management** Supported versions of the configuration items and their relationships are managed in version control.

F - Architecture alignment

A. Software and technical architecture alignment the software architecture of an application is aligned with a technical architecture before a release. **B. Continuous architecture evolvement** the software and technical architecture evolve mutually in a continuous fashion in such a way that these architectures are continuously aligned and kept up to date.

F - Infrastructure

A. Manually provisioned infrastructure infrastructure such as development, test, acceptance and production infrastructure is available and provisioned manually. **B. Partly automatically provisioned infrastructure** A part of the infrastructure between development and production is equivalent in terms of configuration and hardware and some or all environments are provisioned automatically. **C. Automatically provisioned infrastructure** infrastructure between development and production is equivalent in terms of configuration and hardware and provisioned automatically. **D. Managed platform services** platform services (such as a web server and a database server) are preconfigured in the platform and allow for applications being directly deployed, among others, while rights and rolls are managed per environment. This is also known as platform as a service.

References

1. Bekkers, W., van de Weerd, I., Spruit, M., Brinkkemper, S.: A framework for process improvement in software product management. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. CCIS, vol. 99, pp. 1–12. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15666-3_1
2. Nord, R.L., Ozkaya, I., Kruchten, P.: Agile in distress: architecture to the rescue. In: Dingsøyr, T., Moe, N.B., Tonelli, R., Counsell, S., Gencel, C., Petersen, K. (eds.) XP 2014. LNBIP, vol. 199, pp. 43–57. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-14358-3_5
3. Derniame, J.-C., Kaba, B.A., Wastell, D. (eds.): Software Process: Principles, Methodology, and Technology. LNCS, vol. 1500. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-49205-4
4. Erich, F., Amrit, C., Daneva, M.: Cooperation between software development and operations: a literature review. In: The 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ACM, Torino (2014)
5. de Feijter, R., van Vliet, R., Jagroep, E., Overbeek, S., Brinkkemper, S.: Towards the adoption of DevOps in software product organizations: a maturity model approach. Technical report, Utrecht University (2017)
6. Hall, R.S., Heimbigner, D., van der Hoek, A., Wolf, A.L.: An architecture for post-development configuration management in a wide-area network. In: The 17th International Conference on Distributed Computing Systems, pp. 269–278. IEEE, Baltimore (1997)
7. Heitlager, I., Jansen, S., Helms, R., Brinkkemper, S.: Understanding the dynamics of product software development using the concept of coevolution. In: The 2nd international workshop on Software Evolvability, pp. 16–22. IEEE, Philadelphia (2006)
8. Hevner, A., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**, 75–105 (2004)
9. Humble, J., Farley, D., Spafford, G.: Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation. Pearson Education, Boston (2010)
10. Iden, J., Tessem, B., Päivärinta, T.: Problems in the interplay of development and it operations in system development projects: a delphi study of norwegian it experts. Inf. Softw. Technol. **53**, 394–406 (2011)
11. Kim, G., Behr, K., Spafford, G.: The Phoenix Project: A Novel About It, DevOps, and Helping Your Business Win. IT Revolution, Portland (2014)
12. Laan, S.: It Infrastructure Architecture-Infrastructure Building Blocks and Concepts, 2nd edn. Lulu Press, Raleigh (2013)
13. Lawton, G.: Developing software online with platform-as-a-service technology. Computer **41**, 13–15 (2008)
14. Lwakatare, L.E., Kuvaja, P., Oivo, M.: Dimensions of DevOps. In: Lassenius, C., Dingsøyr, T., Paasivaara, M. (eds.) XP 2015. LNBIP, vol. 212, pp. 212–217. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18612-2_19
15. Onwuegbuzie, A.J., Leech, N.L., Collins, K.M.: Qualitative analysis techniques for the review of the literature. Qual. Rep. **17**, 1–30 (2012)
16. Pahl, C., Xiong, H., Walshe, R.: A comparison of on-premise to cloud migration approaches. In: Lau, K.-K., Lamersdorf, W., Pimentel, E. (eds.) ESOCC 2013. LNCS, vol. 8135, pp. 212–226. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40651-5_18

17. Saldaña, J.: The Coding Manual for Qualitative Researchers. Sage, Thousand Oaks (2015)
18. Smeds, J., Nybom, K., Porres, I.: DevOps: a definition and perceived adoption impediments. In: Lassenius, C., Dingsøyr, T., Paasivaara, M. (eds.) XP 2015. LNBIP, vol. 212, pp. 166–177. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18612-2_14
19. Steenbergen, M., Bos, M., Brinkkemper, S., van de Weerd, I., Bekkers, W.: Improving is functions step by step: the use of focus area maturity models. Scand. J. Inf. Syst. **25**, 35–56 (2013)
20. Waller, G., Ehmke, N., Hasselbring, W.: Including performance benchmarks into continuous integration to enable DevOps. ACM SIGSOFT Softw. Eng. Notes **40**, 1–4 (2015)
21. van de Weerd, I., Bekkers, W., Brinkkemper, S.: Developing a maturity matrix for software product management. In: Tyrväinen, P., Jansen, S., Cusumano, M.A. (eds.) ICSOB 2010. LNBIP, vol. 51, pp. 76–89. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13633-7_7
22. Westfechtel, B., Conradi, R.: Software architecture and software configuration management. In: Westfechtel, B., van der Hoek, A. (eds.) SCM 2001/2003. LNCS, vol. 2649, pp. 24–39. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39195-9_3
23. Wettinger, J., Andrikopoulos, V., Leymann, F.: Enabling DevOps collaboration and continuous delivery using diverse application environments. In: Debruyne, C., Panetto, H., Meersman, R., Dillon, T., Weichhart, G., An, Y., Ardagna, C.A. (eds.) OTM 2015. LNCS, vol. 9415, pp. 348–358. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26148-5_23
24. Wettinger, J., Breitenbücher, U., Leymann, F.: Compensation-based vs. convergent deployment automation for services operated in the cloud. In: Franch, X., Ghose, A.K., Lewis, G.A., Bhiri, S. (eds.) ICSOC 2014. LNCS, vol. 8831, pp. 336–350. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45391-9_23
25. Wieringa, R.J.: Design Science Methodology for Information Systems and Software Engineering. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43839-8
26. Yin, R.K.: Case Study Research: Design and Methods. Sage Publications, Thousand Oaks (2013)

An Agile Modeling Oriented Process for Logical Architecture Design

Nuno Santos[1,2](✉), Jaime Pereira[1,2], Francisco Morais[1,2], Júlio Barros[1,2], Nuno Ferreira[2,3], and Ricardo J. Machado[1,2]

[1] CCG/ZGDV Institute, Guimarães, Portugal
nuno.santos@ccg.pt
[2] School of Engineering, ALGORITMI Centre, University of Minho, Guimarães, Portugal
[3] I2S Insurance Knowledge, SA, Porto, Portugal

Abstract. Stakeholders are crucial participants for eliciting requirements towards a new software solution. However, agreeing a common understanding among them is a complex task in a project's initial phase when solution requirements and design need to be refined and/or are unknown. In order to not delay the initial phase and thus endanger the entire project, this paper proposes a process to elicit functional requirements and to design a candidate logical architecture (*i.e.*, without refining information), called Agile Modeling Process for Logical Architectures (AMPLA). By eliciting a set of "just-enough" UML Use Cases, *i.e.*, that includes at least the core requirements information, it is proposed the use of a logical architecture derivation method, the *Four-Step-Rule-Set* (4SRS). This approach is suitable in agile software development contexts, where the solution's architecture is not known upfront.

Keywords: Agile modeling, agile RE · Incremental design
Logical architectures · "Just-enough" architecture

1 Introduction

Companies often strive to properly perform requirements engineering (RE) tasks in software solutions for complex ecosystems (mainly those related to the emergence of new paradigms like Cloud Computing and more recently Industry 4.0, Internet of Things (IoT), machine-to-machine, cyber-physical systems, etc.). The elicitation for the required functionalities regarding the adoption of these recent technologies typically ends up without consensus when technical decisions are required. This trend does not have yet mature references and standards that companies may blindly follow, so the product development results in refactoring efforts towards new architectural patterns.

Stakeholders must able to communicate in what way a future solution improves their business, by defining the product roadmap. A product roadmap is an initial high level project scope and direction [1]. Typically, a first release on a new product encompasses a product's subset able to address priority scenarios, previously identified in order to respond to market needs. In fact, many of these product releases are market–driven, where the release is deployed into the market so it is possible to get feedback from it, *i.e.*, a minimum viable product (MVP). Alongside with these requirements concerns, projects struggle to design candidate architectures for the MVP, endangering development when they conclude that the architecture requires modifications and updates. In an era where software development is more and more agile-oriented, the upfront effort is replaced by the emergence of the design throughout iterative cycles. Such efforts are in opposition to of "*Big Design Upfront*" (BDUF). In agile software development (ASD) contexts, BDUF approaches often result in features that are disregarded after some time ("*You Aren't Gonna Need It*", YAGNI, features).

This paper proposes an Agile Modeling Process for Logical Architectures (AMPLA), an approach for supporting the emergence of a candidate (logical) architecture, rather than BDUF the architecture in an early phase. AMPLA includes the core known features within the initial phase and designs a logical architecture using a stepwise method, without refining information. The emerging characteristics of AMPLA are supported in four stages, two performed before development cycles or Sprints and two in parallel with ASD cycles: (1) eliciting a small set of high-level requirements (see Sect. 2.1); (2) deriving a candidate logical architecture (see Sect. 2.2); (3) define subsystems for refinement (see Sect. 3.1); and (4) refine requirements and the architecture regarding the subsystem in small cycles or Sprints (see Sect. 3.2).

As the name implies, AMPLA is guided by the Agile Modeling (AM) [2] philosophy. AM is about modeling practices, aiming to deliver small portions of models and collecting feedbacks, within ASD processes. AMPLA is driven to be "lean", because it aims to minimize waste within modeling, since the emerging modeling of the features enables leaving out YAGNI features. AMPLA is based in UML models, namely Use Cases diagrams for requirements, and Components diagrams for the logical architecture design. A logical architecture is a view that primarily supports the functional requirements, taken mainly from the problem domain [3]. For the architecture derivation, AMPLA uses the *Four-Step-Rule-Set* (4SRS) method [4] in order to assure the logical components are aligned with the functional requirements. AMPLA is demonstrated in this paper within a research project called Unified Hub for Smart Plants (UH4SP).

This paper is structured as follows: Sect. 2 refers to a "just-enough" requirements modeling of MVP functionalities and the derivation of the candidate architecture; Sect. 3 addresses the architecture modularization, and the models refinement for each module, in order to be implemented within Sprints; Sect. 4 describes the application of the approach in a research project; Sect. 5 presents a comparison with related work; and Sect. 6 presents the conclusions.

2 The Modeling Process for the Candidate Architecture

2.1 Elicitation of "Just-Enough" Requirements

In this section, the objective is to describe the elicitation of the core requirements, and additionally to include techniques for deciding when the "just-enough" requirements model is complete. The elicitation of "just-enough" requirements, rather than promoting their elicitation all upfront, typically is due to insufficient knowledge about using recent technologies. Not only there is an inclusion of a new technology, and their unpredictable adoption, but also of new business models, processes, and the role of stakeholders within the supply chain.

The business needs, project goals, vision document, and other information that act as inputs for requirements is gathered in the inception phase (or earlier) of the project. In this phase, and namely in agile context, where the requirements are not known upfront, it is very difficult to know in advance how much RE is "just-enough". The vision document, for instance, reflects stakeholder's intentions (*i.e.*, features) towards the entire product, however the main purpose at this time is to assure that such features are included. At this point, stakeholders have decided which features to include in the MVP. The requirements that are object of "just-enough" refinement and modeling relate to such features, while the remaining features from the product roadmap may be refined afterwards.

MVP requirements elicitation may be performed by gathering stakeholder's individual needs, also referred as 'bespoke RE'. Alternatively, it may be oriented to a combination of a number of known customers, or to a mass-market where customers cannot be clearly pinpointed, also referred as 'market-driven RE' (MDRE) [5]. AMPLA is oriented for requirements elicitation together with a set of key stakeholders (e.g., business owner, product lead, etc.) that have clearly defined their business needs, based on a previously defined market strategy.

AMPLA starts by eliciting high-level requirements for the MVP. Key stakeholders are interviewed in order to list a set of their expectations towards the solution (as in "*I expect that the solution is able to do this, and that…*") and their perceived importance, namely to depict the scenarios with highest priority for this release. The interviewees may also identify features of the product roadmap to be included in further releases. The "just-enough" subset should include all identified features, however only the features related to the MVP are object of decomposition. For instance, if a roadmap contains fifty features and only ten are to be implemented in the MVP, the high-level architecture should clearly support those ten, but also include initial support for the upcoming forty features.

This paper illustrates the elicitation process based on stakeholders' expectations. However, every business requirements-related information or document that provide inputs for the software requirements elicitation are useful for validating if high-level requirements were considered. Techniques like interviews, questionnaires, workshops, etc., are additional and complemental approaches of the aforementioned document analysis for gathering inputs on requirements. The use of these techniques is a decision of the requirements engineers as they best fit in a given context.

The elicitation process' goal is to model functional requirements in UML Use Case diagrams, promoting the modeling of the product roadmap by functional decomposition, in compliance with a work-breakdown structure (WBS). Use cases are decomposed once or twice, instead of several times like in upfront contexts. Since the main idea is to model "just-enough" requirements, one decomposition may be sufficient, namely the ones that stakeholders are aware at this point. The use of UML Use Case diagrams is mandatory in this approach, since the 4SRS method for deriving the logical architecture requires Use Cases as input.

To validate if the modeled use cases cover the requirements defined in the product scope, Table 1 exemplifies the crosschecking between the stakeholders' expectations and the project goals. Since the expectations and goals list emphasizes the MVP features, there is a context for MVP features to be more decomposed than the remaining. The premises is that if all these concerns are included in the Use Case model, with more emphasis in decomposition detail to MVP requirements, one may consider that we have "just-enough" requirements information for this stage.

Table 1. Traceability matrix of requirements within the initial expectations

Req.	Expec. 1	Expec. 2	Expec. 3	Expec. n
UC.1	x			
UC.2		x		
UC.3			x	
UC.n	x			x

Like in any requirements process, one of the first critical tasks is to identify all projects stakeholders, as well as the solution's interacting actors. By mapping stakeholders to the use cases (Table 2), one must assure that every stakeholder/actor has at least a requirement mapped to it, or is a symptom that critical requirements are missing.

Table 2. Traceability matrix of requirements within the identified project stakeholders and solution actors

Req.	Stkh A	Stkh B	Stkh C	Stkh D
UC.1	x			
UC.2		x	x	
UC.3				x
UC.n	x			

The mappings from Tables 1 and 2 validates that the UML Use Cases diagram includes the features for MVP but also the product roadmap, and that all stakeholders have related functionalities. Both tables provide the required traceability to the expectations and stakeholder/actor needs, which hence are the mechanism used to respond to changes. When applied to mid-and long term projects, the approach supports updates of expectations and stakeholder/actor needs over time, as well as the inclusion of new Use Cases, by tracing requirements in Table 1. Additionally, the approach is able to lead with organizational changes, by tracing the Use Cases to stakeholder/actor needs in Table 2. The approach differs from the segmentation of requirements into different priorities by setting an initial set of Use Cases to the entire solution, refining subsets of the model (Sect. 3.2) and prioritizing them before actually begin the implementation.

Having all the “just-enough” requirements elicited, gathered, modeled and validated, these Use Cases are now able to be used as input for the candidate logical architecture derivation, composed with the “just-enough” architectural components.

2.2 Candidate Architecture Design with 4SRS

The 4SRS method [4] is composed by transformation steps that support mapping between the requirements models (using UML use cases) and architectural model composed by UML components. The usage context of the proposed 4SRS, with “just-enough” requirements and deriving the logical architecture with “just-enough” components is depicted in Fig. 1. Previous experiences with the 4SRS method relate do BDUF contexts, where a larger number of requirements were known upfront, rather than executing the method with smaller number of requirements. However, since in AMPLA the requirements will be later refined and will emerge, the 4SRS method is regularly revisited alongside the development Sprints.

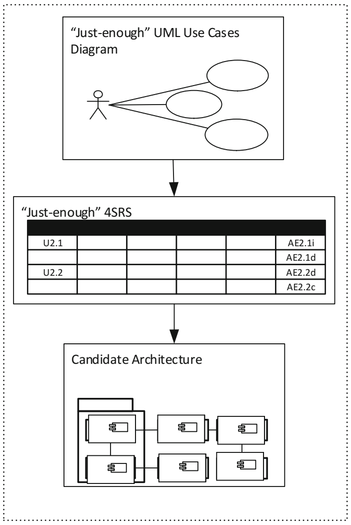


Fig. 1. Method for designing the candidate architecture with 4SRS.

The 4SRS method is composed by four steps: Component Creation; Component Elimination; Packaging and Aggregation; and Component Associations.

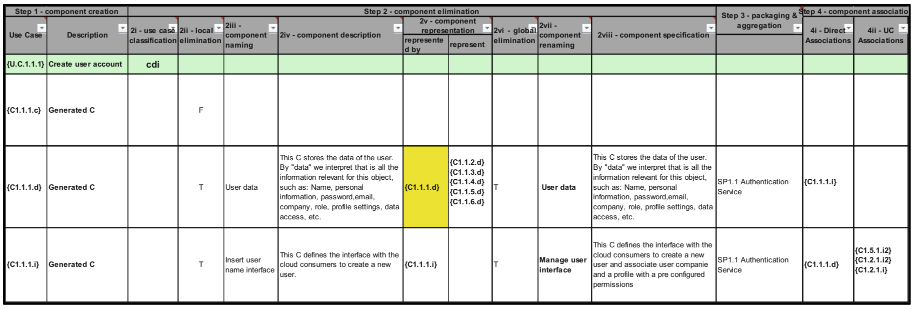
The first step regards the creation of software architectural components. The 4SRS method associates, to each component found in analysis, a given category [6]: interface, data, control. Interface components refer to interfaces with users, software or other entities (*e.g.*, devices, etc.); data components refer to generic repositories, typically containing the type of information to be stored in a database; and control components refer to the business logic and programmatic processing. This categorization makes the architectures derived by the 4SRS to be compliant with architectures from object-oriented programming, or by Model-View-Controller (MVC) patterns.

In the second step, components are submitted to elimination according to pre-defined rules. At this moment, the system architect decides which of the original three components (i, c, d) are maintained or eliminated, firstly taking into account the context of a use case from Step 1, and later compared for redundancy within the entire system. Additionally, each component is named and textually described relating to its behavior.

In the third step, the remaining components (those that were maintained after executing step 2), where there is an advantage in treating them in a unified process, should give the origin to aggregations or packages of semantically consistent components.

The final step refers to the associations between components. The method provides steps for identifying such associations based in descriptions from use cases, as well as from the own components during the components creation.

The execution of the 4SRS transformation steps can be supported in tabular representations. These representations enables partial automation of the transformations steps and constitute the main mechanism to automate a set of decision assisted model transformation steps. A small part of the 4SRS method execution table is represented in Fig. 2. The table is not zoomed due to paper size limitations. The cells are filled with the set of decisions that were taken and made possible the derivation of a logical architecture. Each column of the table concerns a step/micro-step of the method execution.



Step 1 - component creation		Step 2 - component elimination									Step 3 - packaging & aggregation	Step 4 - component associatio	
Use Case	Description	2i - use case classification	2ii - local elimination	2iii - component naming	2iv - component description	2v - component representation		2vi - global elimination	2vii - component renaming	2viii - component specification		4i - Direct Associations	4ii - UC Associations
						represented by	represent						
{U.C.1.1.1}	Create user account	cdi											
{C1.1.1.c}	Generated C		F										
{C1.1.1.d}	Generated C		T	User data	This C stores the data of the user. By "data" we interpret that is all the information relevant for this object, such as: Name, personal information, password,email, company, role, profile settings, data access, etc.	{C1.1.1.d}	{C1.1.2.d} {C1.1.3.d} {C1.1.4.d} {C1.1.5.d} {C1.1.6.d}	T	User data	This C stores the data of the user. By "data" we interpret that is all the information relevant for this object, such as: Name, personal information, password,email, company, role, profile settings, data access, etc.	SP1.1 Authentication Service	{C1.1.1.i}	
{C1.1.1.i}	Generated C		T	Insert user name interface	This C defines the interface with the cloud consumers to create a new user.	{C1.1.1.i}		T	Manage user interface	This C defines the interface with the cloud consumers to create a new user and associate user companie and a profile with a pre configured permissions	SP1.1 Authentication Service	{C1.1.1.d}	{C1.5.1.i2} {C1.2.1.i2} {C1.2.1.i}

Fig. 2. 4SRS method execution using tabular transformations

The output is a logical architectural diagram, composed by a set of software components and relations/flows between them. The architectural diagram is modeled in UML Components, as depicted in a simple example in Fig. 3.

It must be pointed out that AMPLA and the 4SRS supports the logical view [3] of the architecture, namely the identification and design of software components referring to functional requirements. Architecture design should also address quality requirements. Addressing quality requirements is out of the scope of AMPLA, but is able to coexist with other architecture-centric methods such as QAW or ADD.

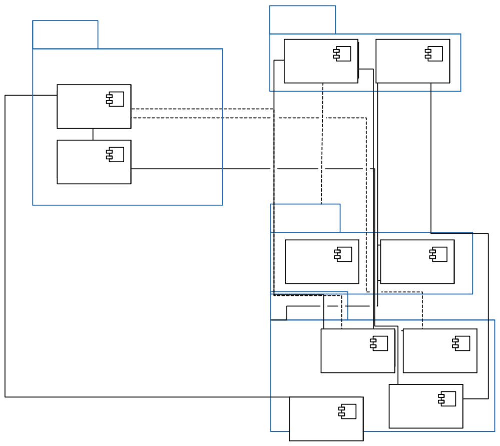


Fig. 3. Simple example of a candidate UML components architecture

The use of 4SRS throughout the process, first in the scope of the candidate architecture, and afterwards in the scope of each refinement, provides the traceability between components and the functional requirements, allowing an agile response to changing requirements. The number of components or the number of associations between components are possible indicators to determine the architecture's riskiest components.

3 Incremental Design for Refining the Logical Architecture

3.1 Subsystem Bordering

The idea is to incrementally refine the architecture, in parallel with the implementation efforts regarding the architectural components that already were refined. Like in most cases, the best approach to issue complex problems is to divide them into smaller ones and address them one by one, ultimately allowing to address the big solution.

With the purpose of modularizing the architecture, the logical architecture from Fig. 3 is partitioned into sub-systems. As previously mentioned in Sect. 1, AMPLA is able to be performed in parallel within a typical ASD context, however the subsystem bordering is helpful in complex ecosystems contexts. This bordering is likely to reflect the ecosystem, e.g., IoT, cloud, infrastructure, embedded systems, etc., and the dependencies between the subsystems. The sub-system border provides context for modeling more refined Use Cases, as well as context for making technical design decisions that in the initial phase were very difficult to make. The Use Cases may then be used for a new execution of the 4SRS (Fig. 4), since they relate to "new" Use Cases that emerged and hence did not exist before the definition of the sub-systems. Within these "new" Use Cases, they may relate to refined use cases from the previous model, as well as relating to new use cases within the same level of refinement. *E.g.*, a sub-system related to a use case {UC1.5} may result in refinements like {UC1.5.1} and {UC1.5.2}, as well as new use cases in the same level of refinement like {UC1.6} and {UC1.7}.

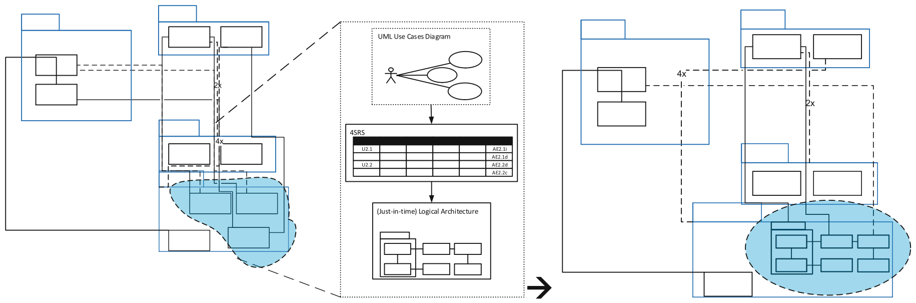


Fig. 4. Recursive execution of 4SRS for refining a given example module.

By performing the 4SRS method within this new use case model, the resulting software components relate to refined components for that sub-system, that can be "replaced" in the initial candidate architecture, resulting in a more refined version. In order to assure that the defined sub-system, even though composed by more components, is still able to fit in the overall logical architecture, its interfaces must be maintained (system of systems theory). It must be pointed out that the execution of the 4SRS for the new and emerged use cases in Fig. 4 are an increment to the previous execution, rather than performing a new 4SRS execution with only the new use cases.

3.2 Subsystem Refinement and Implementation

The sub-system requirements are object of analysis within each iterative cycle. In this section is described the refinement process aligned with iterative cycles, e.g., Scrum Sprints. Figure 5 depicts the distribution of the module requirements for refinement and respective implementation tasks to be included as Product Backlog Items (PBI), distributed within the cycles (Sprints).

After each subsystem is defined, the process is then structured as an iterative approach (e.g., Scrum Sprints). Within every iteration, the team performs tasks involving several software engineering disciplines. This paper uses the terminology from RUP's and AUP's disciplines (only for demonstration purposes) to depicts the type of effort involved within the Sprints. These efforts are illustrated in Fig. 5 by the colored bars within each cycle, where each bar is a software engineering disciplines, and with more or less effort during the cycle, similar to RUP and AUP. The main difference is that, in parallel with carrying out typical disciplines within the Sprints (Implementation, Testing and Deploy) that result in the delivery of a software increment, other team members are responsible for refining requirements regarding the features not yet included in the team's Sprint Backlog, and that are planned to be implemented in further Sprints. These requirements are modeled, hence the Requirements discipline, and then an incremental execution of the 4SRS method, hence the Analysis & Design (A&D) discipline.

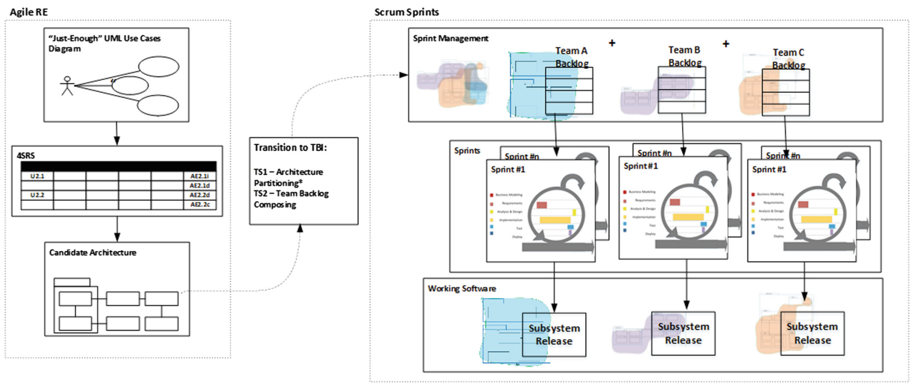


Fig. 5. Distributed implementation of each architecture module

The Requirements and A&D tasks are performed in iterative cycles and incrementally, synchronized with the development and deployment during the Sprints, in a sense that what is modeled during a cycle is then ready to be implemented in next cycles (Fig. 6). While the implementation is being performed, requirements from the original logical architecture, not yet refined in previous cycles, are modeled in use cases. They originate a new increment of the 4SRS method, deriving additional components. These models require validation from customer before being ready to be included in the Team Backlog. The first time this requirements cycle is being performed, the development Sprint is not yet performed (or a Sprint 0 cycle is performed where no software is delivered, but rather the development infrastructure is built). Then, requirements and the 4SRS address the functionalities to be implemented in the next Sprint or even more ahead Sprints. It is assumed that, since the components have been refined, they are in an improved situation to be now passed for implementation.

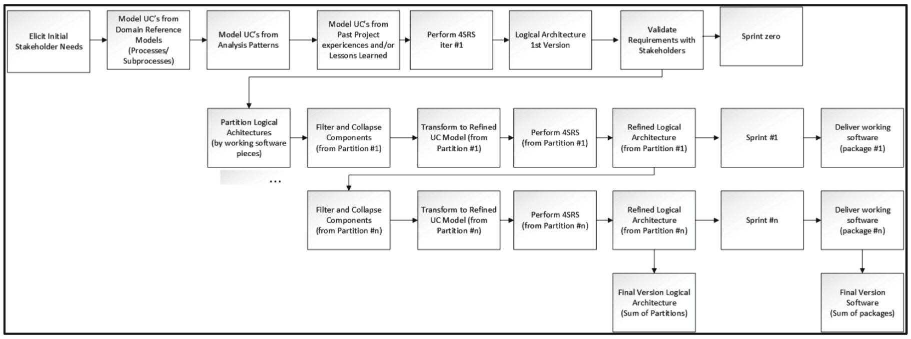

Elicit Initial Stakeholder Needs
Model UC's from Domain Reference Models (Processes/ Subprocesses)
Model UC's from Analysis Patterns
Model UC's from Past Project expericences and/or Lessons Learned
Perform 4SRS iter #1
Logical Architecture 1st Version
Validate Requirements with Stakeholders
Sprint zero
Partition Logical Achitectures (by working software pieces)
Filter and Collapse Components (from Partition #1)
Transform to Refined UC Model (from Partition #1)
Perform 4SRS (from Partition #1)
Refined Logical Architecture (from Partition #1)
Sprint #1
Deliver working software (package #1)
...
Filter and Collapse Components (from Partition #n)
Transform to Refined UC Model (from Partition #n)
Perform 4SRS (from Partition #n)
Refined Logical Architecture (from Partition #n)
Sprint #n
Deliver working software (package #n)
Final Version Logical Architecture (Sum of Partitions)
Final Version Software (Sum of packages)

Fig. 6. Incremental requirements and 4SRS execution throughout the Sprints

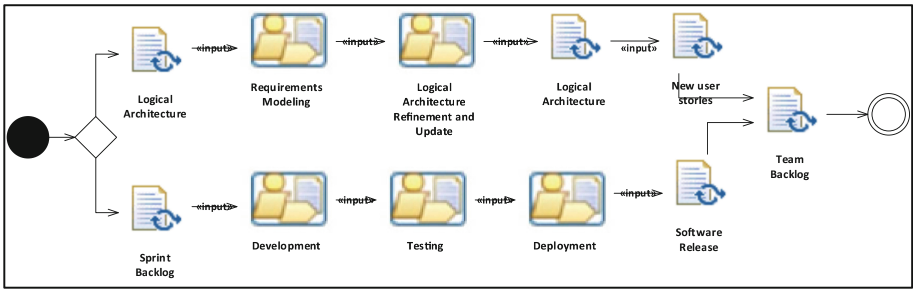


Fig. 7. Parallel tasks within Sprints in SPEM diagram

While this refined information is implemented, coded, tested and deployed, ending in a delivery of a software increment to customer, in parallel other model is refined within the same approach (*i.e.*, use cases, 4SRS and architecture). This process is depicted in Fig. 7 as a Software Process Engineering Metamodel (SPEM) diagram. This way, when a cycle is finished, it is expectable that new sub-systems were refined and able for implementation.

4 Applying the Method in the UH4SP Research Project

The UH4SP project aims developing a platform for integrating data from distributed industrial unit plants, allowing the use of the data acquired from IoT systems for enterprise-level production management and collaborative processes between plants, suppliers, forwarders and clients, which are supported by software applications and services deployed in a cloud platform. The UH4SP services were validated within the project by performing a set of pilot scenarios, mostly related to the cement domain. Since IoT, cloud computing and Industry 4.0 are quite recent and still without a *de facto* standard for a reference model to lean on, the project started by eliciting user requirements for tackling the business processes, without refining technical requirements whether regarding cloud and IoT systems.

This section describes the requirements elicitations, modeling, candidate logical architecture derivation, refinement and delivery of the backlog items. The project's objectives are used as input for the high-level use case modeling were: (1) to define an approach for a unified view at the corporate (group of units) level; (2) to develop tools for third-party entities; (3) in-plant optimization; and (4) system reliability. The requirements elicitation started by listing a set of stakeholder expectations towards the product roadmap, encompassing the entire product but only MVP features were detailed. The expectations list of the project included 25 expectations, categorized by environment, architecture, functional and integration issues. They relate to business needs that afterwards allowed depicting functional requirements, modeled in use cases (Fig. 8).

The Use Case model was globally composed by 37 use cases after the decomposition. Use case *{UC.1} Manage business support* was decomposed in five use cases,

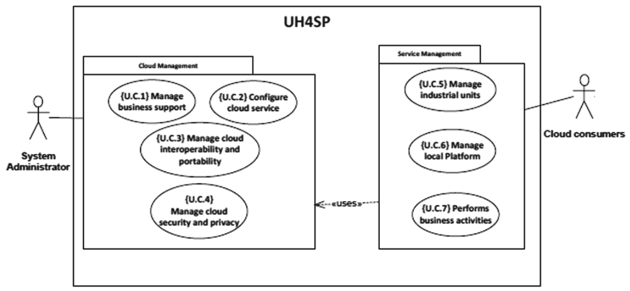

UH4SP
Cloud Management
{U.C.1} Manage business support
{U.C.2} Configure cloud service
{U.C.3} Manage cloud interoperability and portability
{U.C.4} Manage cloud security and privacy
Service Management
{U.C.5} Manage industrial units
{U.C.6} Manage local Platform
{U.C.7} Performs business activities
«uses»
System Administrator
Cloud consumers

Fig. 8. UH4SP first-level Use Cases.

use case *{UC.2} Configure cloud service* was decomposed in eight use cases, use case *{UC.3} Manage cloud interoperability and portability* was decomposed in five use cases, use case *{UC.4} Manage cloud security and privacy* was decomposed in three use cases, use case *{UC.5} Manage industrial units* was decomposed in two use cases, use case *{UC.6} Manage local Platform* was decomposed in five use cases, and use case *{UC.7} Performs business activities* was decomposed in ten use cases. Almost the entire model was detailed in one lower-level (*e.g.*, {UC5.1}, {UC5.2}, etc.). Only the cases of *{UC.1} Manage business support*, *{UC.2} Configure cloud service* and *{UC.7} Performs business activities* included an additional decomposition, composed with three use cases each, and are examples of bigger sized features of the MVP (based in the quantity of low-level use cases). Use cases *{UC.3} Manage cloud interoperability and portability* and *{UC.4} Manage cloud security and privacy* relate to features not addressed in the MVP, hence were not object of further decomposition. The total of 37 use cases perceive the low effort in decomposing at this phase, taking into account the large-scale nature of the project, namely the number of expectations (25) and that it is to be implemented by five separate teams.

The UH4SP logical architecture had as input 37 use cases and, after executing 4SRS method, was derived with 77 architectural components that compose it (Fig. 9). It is not the purpose of this diagram in this paper to analyze its components, but rather to have a representation to follow the incremental design, thus the figure is not zoomed and its components' name are not properly readable. This architecture was afterwards divided in a set of modules to be assigned to each of the project's teams (Fig. 10).

The modularization depicted in Fig. 10 originated 5 modules/subsystems, each assigned for 'Team A', 'Team B', 'Team C', 'Team D' and 'Team E'. The bordering was based in the contributions that each team brings to the consortium, namely IoT, cloud infrastructure, cloud applications and sensors/embedded systems. Each border is thus a part of a complex ecosystem and the dependencies are depicted in Fig. 10. 'Team B' was the focus of this research. During the first "Just-enough" modeling, there were 37 use cases and 77 architectural components after performing the 4SRS method. After modularizing, 11 use cases from the 37, and the 15 components from the 77, compose the module under analysis. Finally, the requirements refinement output 29 use cases, *i.e.*, 18 refined functionalities, which then derived 37 architectural components from the 4SRS method. Having in mind the large-scale and complex ecosystem context

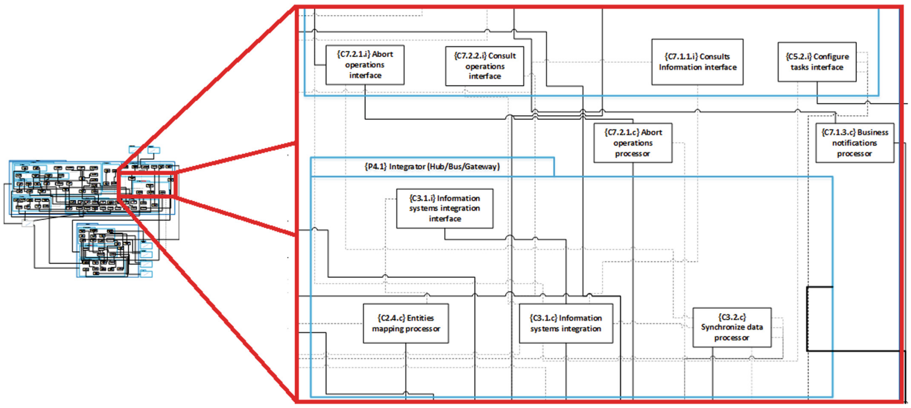

{C7.2.1.i} Abort operations interface
{C7.2.2.i} Consult operations interface
{C7.1.1.i} Consults Information interface
{C5.2.i} Configure tasks interface
{C7.2.1.c} Abort operations processor
{C7.1.3.c} Business notifications processor
{P4.1} Integrator (Hub/Bus/Gateway)
{C3.1.i} Information systems integration interface
{C2.4.c} Entities mapping processor
{C3.1.c} Information systems integration
{C3.2.c} Synchronize data processor

Fig. 9. UH4SP logical architecture derived after 4SRS execution.

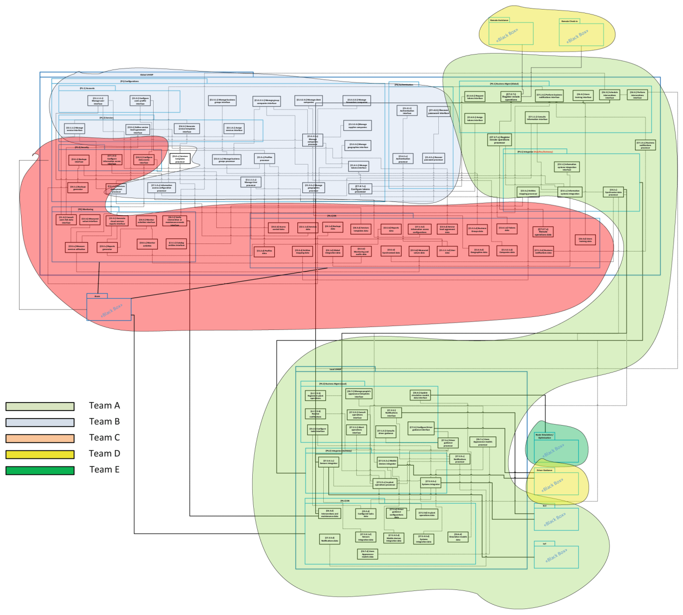


Fig. 10. The modularization of the logical architecture

of the project, these values may be perceived as acceptable. It is an increase of almost three times as the original models. The refinement was performed incrementally and in parallel with team's Sprints (using Scrum), as in Fig. 6. The requirements were refined, modeled and validated with the consortium, and only afterwards were input for the 4SRS method. The requirements validation was thus iterative as well. In UH4SP, after modeling in UML Use Cases, the requirements package was also composed with wireframes, to enrich the discussion. Additionally, in Scrum's Sprint Retrospective ceremonies, these models were object of feedback, and, if applicable, missing requirements were included. These validations were crucial in the project to enable a complete team buy-in. After the 4SRS method, those five use cases derived 11 components. The MVP was implemented at the end of these Sprints, which was composed by 94 components. These components supported the project's pilot scenarios. However, next releases in order to follow the product roadmap are to be developed.

5 Comparison with Related Work

The requirements discipline is no longer performed only in the early phase of the project, but rather in a continuous way [1, 7]. In ASD contexts, where widely known frameworks, such as Scrum, XP, Kanban, Crystal, etc. don't promote extensive use of requirements models (*e.g.*, UML), modeling tasks are able to be included if performed continuously and incrementally, as stated by Ambler's "Agile Modeling" [2]. In the same line of reasoning, Ambler presents an evolution and emerge-oriented approach for using models in ASD, called agile model-driven development (AMDD) [8] point out that a project can start with "just-enough" requirements and "just-enough" architecture and then continuously specify requirements and update this so-called candidate architecture design alongside implementation efforts (e.g., Scrum Sprint cycles). In these contexts, the architecture design must assure that at least the high-level information is included, providing the organization of high-level software components. These components do not have to be refined for implementation purposes, but rather to provide a first insight on the structure of the aimed solution. AMPLA is aligned with this reasoning, however AMDD does not state any method for assuring the alignment of the architecture with the elicited needs, which may be assured in AMPLA.

In this agile era, even the project's early phase is intended to be as short as possible. The inception, like the pregame phase or Sprint zero in Scrum, aims providing a shared understanding of the project and the required information for the development phase. The lean inception [9] relates to lean startup (and MVPs) contexts, where the product scope is characterized as quickly as possible. In AMPLA this reasoning is also applicable, since actors, features and architecture are defined as soon as possible. Alternatively, elicitation could be based in involving customers or other market entities (MDRE). This context is not included in this paper, but will be object of future research.

Minimize the waste is a universal concern in every project, regardless its nature. Specifically, within software architecture, some approaches aiming agility and lean architecting intend to leave out YAGNI features in the minimal time possible. Clements *et al.* state "*If information isn't needed, don't document it*" [10]. All documentation should have an intended use and audience in mind, and be produced in a way that serves both. artifacts could include specifications written in UML or an architecture description language if appropriate, but a few informal box-and-line diagrams, descriptions of a system metaphor, a succinct document capturing the relevant decisions, and combinations thereof might do the job as well or better [11].

In general, the set-up phase of the project - often called '*iteration 0*' [12] - includes some upfront design, with ongoing architectural refinement during the iterative development. However, how can an architect or developer determine what the correct amount of "just enough up-front design" is, *i.e.*, how can agile teams reduce the up-front effort without sacrificing the benefits of an up-front design [13]? This work states concepts for reducing the amount of effort in up-front decision making such as "*using predefined architecture*", "*intuitive architecture*", "*having architectural experience simplifies decision making*" and "*being familiar with the architecture*".

Coplien and Bjørnvig present the concept of "Lean Architecture" [14], where the lean perspective focuses on how to develop the overall system form by drawing on experience and domain knowledge. One of the proposals for performing design as concepts and requirements emerge, included in the research of Abrahamsson [12] and Farhan [15] is the approach of a walking skeleton. Abrahamsson refers to it as an architectural prototype [12]. Farhan refers to it as a tiny implementation of the system that performs minimum functionality. Kazman proposes the design of a candidate architecture [16]. Cockburn proposes, for larger projects with unstable requirements, to start by quickly designing a candidate architecture even if it leaves out many details. [17]. He considers an approach to create a simple architecture that handles all the known "*big rocks*" (requirements that are particularly hard to incorporate late in the project), and handle the "small rocks" as they appear during the project. These works describe some strategies for using a candidate architecture approach and its evolution. However, these approaches lack in guiding the architecture design and evolution process aiming the alignment with the gathered requirements.

Architecture approaches are also present in ASD, where architecture-centric methods such as QAW, ADD, ATAM/CBAM and ARID are performed in parallel with the development iterations [15, 18–22]. Costa *et al.* propose a technique for deriving User Stories statements from architectures that resulted from the 4SRS method execution [23]. AMPLA is able to be included in these frameworks, since all of them use architecture as basis for implementation decisions, and includes a stepwise method for assuring proper alignment of the logical architecture with the elicited requirements.

6 Conclusions and Future Work

Especially when new technological paradigms arise, stakeholders have many difficulties in eliciting technical design decisions. In opposition to waterfall-based frameworks, where all the requirements and design tasks are performed only upfront, within ASD projects these tasks should be performed continuously. For that reason, this paper presents the AMPLA approach. AMPLA provides a method for deriving a candidate logical architecture based in UML Use Cases, the 4SRS method. The approach also validated the coverage of the elicited "just-enough" model, gathered together with identified key stakeholders. The "just-enough" UML Use Case model allows to early identify main features, which provides an overview of the project and its scope. We acknowledge the impact of features' characteristics (size, complexity, interconnections, dependencies between subsystems etc.) to management issues, as tasks planning, budget proposals, and resource and skills allocation, which will be later addressed.

The design of "just-enough" architecture used an architectural method, the 4SRS, to derive the candidate architecture based in the small set of ("just-enough") UML Use

Cases. There is not any difference within the steps of the method, in comparison with the original method, to derive a candidate architecture. Rather, as the input are high-level requirements in opposition to more refined ones, one may experience difficulties in identifying a proper classification of the use case in order to decide the components to be maintained within the second step, since a more refined information helps in better define the component's nature. However, as in AMPLA the requirements will be later refined and will emerge, the 4SRS method is used as in a "living table" that is opened alongside the development Sprints, rather than a waterfall-based and one-time-execution approach, providing traceability between the requirements and the components in order to agilely respond to changes.

Acknowledgments. This work was developed within the project UH4SP: Unified Hub 4 Smart Plants (Project ID 017871), under Portuguese National Grants Program for R&D projects (P2020 – SI IDT), COMPETE: POCI-01-0145-FEDER-007043 and FCT – Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/00319/2013.

References

1. IIBA: Agile Extension to the BABOK Guide. International Institute of Business Analysis (2013)
2. Ambler, S.: Agile Modeling: Effective Practices for Extreme Programming and the Unified Process. Wiley, New York (2002)
3. Kruchten, P.: The 4 + 1 view model of architecture. IEEE Softw. **12**, 42–50 (1995)
4. Machado, R.J., Fernandes, J.M., Monteiro, P., Rodrigues, H.: Refinement of software architectures by recursive model transformations. In: Münch, J., Vierimaa, M. (eds.) PROFES 2006. LNCS, vol. 4034, pp. 422–428. Springer, Heidelberg (2006). https://doi.org/10.1007/11767718_38
5. Regnell, B., Brinkkemper, S.: Market-driven requirements engineering for software products. In: Aurum, A., Wohlin, C. (eds.) Engineering and Managing Software Requirements, pp. 287–308. Springer, Heidelberg (2005). https://doi.org/10.1007/3-540-28244-0_13
6. Machado, R.J., Fernandes, J.M., Monteiro, P., Rodrigues, H.: Transformation of UML models for service-oriented software architectures (2005)
7. Grau, B.R., Lauenroth, K.: Requirements engineering and agile development - collaborative, just enough, just in time, sustainable. International Requirements Engineering Board (2014)
8. Ambler, S.: Agile Model driven development (AMDD). In: XOOTIC Mag., February 2007
9. Caroli, P.: Lean Inception. https://martinfowler.com/articles/lean-inception/
10. Clements, P., Ivers, J., Little, R., Nord, R., Stafford, J.: Documenting Software Architectures in an Agile World. DTIC Document (2003)
11. Erdogmus, H.: Architecture meets agility. IEEE Softw. **26**, 2–4 (2009)
12. Abrahamsson, P., Babar, M.A., Kruchten, P.: Agility and architecture: Can they coexist? Softw. IEEE **27**, 16–22 (2010)
13. Waterman, M., Noble, J., Allan, G.: How much architecture? Reducing the up-front effort. In: AGILE India, pp. 56–59. IEEE (2012)
14. Coplien, J.O., Bjørnvig, G.: Lean Architecture: for Agile Software Development. Wiley, Hoboken (2011)

15. Farhan, S., Tauseef, H., Fahiem, M.A.: Adding agility to architecture tradeoff analysis method for mapping on crystal. In: WRI World Congress on Software Engineering, WCSE 2009, pp. 121–125. IEEE (2009)
16. Kazman, R.: Foreword - bringing the two together: agile architecting or architecting for agile? In: Agile Software Architecture: Aligning Agile Processes and Software Architectures, pp. xxix–xxx (2013)
17. Cockburn, A.: Agile Software Development: the Cooperative Game. Pearson Education, Upper Saddle River (2006)
18. Madison, J.: Agile architecture interactions. Softw. IEEE **27**, 41–48 (2010)
19. Jeon, S., Han, M., Lee, E., Lee, K.: Quality attribute driven agile development. In: 9th International Conference on Software Engineering Research, Management and Applications (SERA), pp. 203–210. IEEE (2011)
20. Nord, R.L., Tomayko, J.E.: Software architecture-centric methods and agile development. Softw. IEEE **23**, 47–53 (2006)
21. Bellomo, S., Kruchten, P., Nord, R., Ozkaya, I.: How to agilely architect an agile architecture. Cut. IT J. **27**(2), 12–17 (2014)
22. Nord, R.L., Ozkaya, I., Kruchten, P.: Agile in distress: architecture to the rescue. In: Dingsøyr, T., Moe, N.B., Tonelli, R., Counsell, S., Gencel, C., Petersen, K. (eds.) XP 2014. LNBIP, vol. 199, pp. 43–57. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-14358-3_5
23. Costa, N., Santos, N., Ferreira, N., Machado, R.J.: Delivering user stories for implementing logical software architectures by multiple scrum teams. In: Murgante, B., Misra, S., Rocha, A.M.A.C., Torre, C., Rocha, J.G., Falcão, M.I., Taniar, D., Apduhan, B.O., Gervasi, O. (eds.) ICCSA 2014. LNCS, vol. 8581, pp. 747–762. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09150-1_55

Exploring the Design Needs for the New Database Era

Noa Roy-Hubara(✉) and Arnon Sturm

Ben-Gurion University of the Negev, Beer Sheva, Israel
nro@post.bgu.ac.il, sturm@bgu.ac.il

Abstract. During the last decade, new database solutions and technologies have emerged. These were developed in order to facilitate the new types of applications and requirements. However, these new improvements also affected the desired properties existing in traditional databases (i.e., relational). Thus, there is a need to better understand the tradeoffs among the various solutions and to support such analysis with design and modeling techniques. In this paper, we review database design approaches and explore the requirements for these.

Keywords: Modeling · Design · Database

1 Introduction

As databases are the foundation of every information system, it is crucial to note that the field has changed tremendously over the last decade. The relational databases were created, matured and dominated for a long time; in fact, it is still one of the top solutions to date[1]. However, many other solutions emerged. These include the object-oriented, the NoSQL and the NewSQL databases. The latter were developed in order to address existing limitations and new needs, such as allowing better performance, handling complex data, allowing data flexibility and data distribution.

However, the new capabilities of NoSQL databases come at the cost of reduced consistency and lack of accepted standards in query languages, APIs, etc. NewSQL databases are a combination of SQL and NoSQL that try to address the disadvantages of both databases types. Nevertheless, NewSQL is still less mature, has less support and does not support a different kind of data. Object-Oriented databases that bloomed in the 1990s, are integrated with object-oriented programming languages; however, this might be a burden when programming in other types of languages. They also lack standards in querying languages[2].

With the rise of the new database technologies, there was a clear need for methods to design them. As part of the relational model maturation process, many methods for its design were created; the most fundamental one is Peter Chen's entity-relationship model (also known as ER diagrams), which was the basis for many extensions; and the UML class diagrams [3, 22]. The other databases that followed might have solved issues that were raised by users that were unsatisfied with the relational databases, but

1 https://db-engines.com/en/ranking.
2 http://www.faculty.rsu.edu/users/c/clayton/www/waide/paper.htm#_Toc38521773.

lacked the advantages that they already enjoyed, and put less emphasis on the design process.

In this work we set the following research question: what are the considerations needed to be taken when selecting and designing databases for the future information systems. Addressing this question we review database design methods and analyze the emerging needs for database modeling and design and further identify existing gaps in current design approaches.

The rest of the paper is structured as follows. To set the ground for the readers, in Sect. 2 we introduce the different database capabilities and analyze their strengths and limitations. In Sect. 3 we review and analyze different design approaches for each database type. In Sect. 4 we discuss the reviewed database design methods and define the desired features in such methods. Finally, in Sect. 5 we summarize our finding and set plans for future research.

2 Background: The Evolution of Database Technology

Various types of databases exist today. These include relational databases, Object-Oriented databases, NoSQL databases, and NewSQL databases. In this section we analyze these database types in light of the several properties that are discussed in the database literature:

Consistency (CON): Consistency means that once data is written, queries that will follow will be able to read that data. When a database is inconsistent two users at the same time might read different data. We also consider transactions and consistency concepts and if and how the databases handle these.

Integrity (INT): Data integrity refers to the accuracy of the data stored in a database[3]. When choosing or reviewing a database we would like to know if the data is reliable and if the database allows for "incorrect" data.

Performance (PER): In the era of Big Data is it is important to know how a system scales in terms of response time. Scaling may be done horizontally (more machines) or vertically (stronger machine).

Query options (QUE): When considering a database we wish to know which query languages are supported. There is a difference in supporting built-in query languages (such as SQL) and APIs when it comes to usability.

Flexibility (FLEX): Flexibility is the database's ability to handle changes. More specifically flexibility refers to the lack of rigid schema that allows flexible data writing.

2.1 Relational Databases

The relational databases save and manage data as a set of tables containing data fitted into predefined categories (or columns). Each row contains a unique instance of data [24]. The relational model is **enforced** by a rigid schema that describes the tables'

[3] https://www.veracode.com/blog/2012/05/what-is-data-integrity.

structure, data types and constraints. The model supports keys. A primary key is the identifier of a row in each table, and must be unique, while a foreign key is a reference to data from another table. Both types of keys may be comprised of one or more columns. The model supports indexing of columns to achieve fast querying.

One of the most important features of the relational databases is their ACID support, which aims at supporting the concept of a transaction. ACID stands for: Atomicity – all tasks are done completely or not at all, Consistency – a transaction leaves the database in a consistent state when it ends, Isolation – transactions do not interfere with one another, and Durability – when the transactions complete the data will persist permanently [26]. This ensures the integrity and consistency of data when it is needed.

Querying the model is done with SQL – Structured Query Language. SQL is a rather simple declarative language that became the standard in the 1980's [5]. Years brought with them many relational solutions: Oracle, Microsoft SQL Server, MySQL, PostgreSQL and such. The solutions may differ in capabilities, but they all support the same relational model and SQL.

The relational databases are most suited for structured data. These do not work well with complex data [3]. The amount of new data is measured in terabytes and petabytes, and relational databases do not scale out well [25, 33]. In addition, the relational database schema is often too rigid and does not support the required flexibility.

2.2 Object-Oriented Databases

Object-Oriented (OO) databases were created to overcome the much touted impedance mismatch between the relational model and object-oriented programming languages [29]. Such databases model data as objects, so the application and the database "speak" the same language [23]. OO databases support ACID properties, as well as OO properties - encapsulation, polymorphism and inheritance[4].

In [23] it is stated that OO databases are still minor player with solid, yet niche markets. When looking at current database rankings, OO databases are at the bottom and rarely used today. This is due to several reasons. One major reason is that in comparison to the simple SQL, OO databases do not support a cohesive, standard query language [23]. Another issue is related to schema changes: since relational databases are typically independent of the actual application changes (deleting, updating or creating tables) are easy. In the OO databases, however, since the database and the application are highly dependent, any change in the database will cause changes in the application code: according to Dare Obasanjo (see footnote 4): "…modifying the schema by creating, updating or modifying a persistent class typically means that changes have to be made to the other classes in the application that interact with instances of that class". Some other reasons include the maturity and experience of relational databases[5].

[4] http://www.25hoursaday.com/WhyArentYouUsingAnOODBMS.html.

[5] https://softwareengineering.stackexchange.com/questions/178119/why-arent-object-oriented-databases-used-as-much-as-relational-databases/178124#answer-178124.

2.3 NoSQL Databases

NoSQL databases had emerged in the last decade as a result of the rise of Big Data. "No" means: Not-Only and implies about the nature of those databases. NoSQL refers to four types of databases: (1) Key-Value databases; (2) Wide-Column Stores; (3) Document Stores; and (4) Graph databases. As mentioned in the previous section, the relational databases did not handle well the new kind of data which is unstructured and is of a large volume. NoSQL databases were developed in order to address this problem and were meant to enhance scalability, availability, and performance. An example of this data is found on Twitter: a site in which users can post thoughts and reflections (aka Tweets). Twitter has millions of users that post millions of times a day, which requires significant data storage, processing, and data analytics capabilities [20].

Transactional support and consistency enforcement are handled differently in NoSQL solutions [15]. Most NoSQL databases do not support ACID but are rather based on the BASE principles [34, 35]. BASE stands for Basically Available, Soft State and Eventually Consistent, which sacrifices strong consistency in exchange for high availability. In 2000, Brewer developed the CAP theorem. CAP stands for Consistency, Availability and Partition Tolerance, and according to Brewer they cannot all co-exist in one system [16]. Only two of the three CAP needs can be met simultaneously. Hence, NoSQL databases are often categorized based on CAP theorem. Most of them support Partition tolerance in order to support Big Data and data distribution over many nodes.

In NoSQL systems, schemas are defined when reading data as opposed to when writing it. Data can be written in any format, yet, in order to access that data, programmers need to be familiar with the way data is modeled. This means that integrity constraints are not supported, which in time may lead to many issues such as mismatched property names within same entities, unrelated relation types, and missing relations/properties. There is no standard query language for NoSQL databases. While some databases implement a declarative query language similar to SQL, it is not mandatory. Instead, these databases work with APIs, which require manual query programming. This may be either fast for simple tasks or time-consuming for others as complex query programming can be time-consuming and challenging [24]. OLAP-style queries require a lot of application code[6] and some organizations have found that NoSQL solutions lead to wasted developer time, due to the inconsistent data [29].

Key-Value Databases

In the key-value databases data is organized as an associative array of entries consisting of key-value pairs [19]. In such databases, the data is usually stored in the main memory so they have a high speed in data processing [34]. Due to their simplicity, they support only get and put operations [14].

Key-value stores are mainly used when there is a need for higher performance, and since they are stored in memory they are very fast [20]. This is at the expense of complex query functionalities. A key-value solution has to have all of the following features: (1) support only for simple data types; (2) search only based on keys

[6] http://dataconomy.com/2015/08/sql-vs-nosql-vs-newsql-finding-the-right-solution/.

(no composite keys); (3) no support for full-text search; and (4) no support for secondary index [20].

Wide-Column Stores

In wide column stores, data is stored in columns rather than rows [17, 33]. A column family is a group of related columns. As opposed to relational databases in which redundancy is frowned upon, column families support de-normalization, though it may cause consistency problems.

According to [34] these databases are more suitable in cases where the write operation is more frequent than the read operation. Column databases are able to store any data structures, have high performance and highly scalable architecture [16, 34]. Most wide column stores are linked to analytical frameworks such as MapReduce [20], which enables fast analytics.

Column stores have a dynamic schema. However, the design is dependent on access paths that define what and how data is to be retrieved. Since data do not have to be de-normalized, data redundancy exists in multiple tables, which requires careful integrity management at the code level [33].

Document Stores

A document in a document-oriented NoSQL database contains data that is de-normalized, semi-structured and stored hierarchically in the form of a key-value pairs (such as JSON, BSON, etc.) [26]. Documents do not need to have a uniform structure [20], and each document has a key. Document stores usually support for secondary indexes that assist in full-text search and retrieval. Relationship between documents can be expressed either by embedding a document in other documents or by referencing to another document. A reference is similar to the concept of a foreign key that is used by relational databases [26].

By definition, like other NoSQL systems, the document stores do not provide ACID transactional properties [20]. Document stores are good for semi-structured data, and as opposed to the wide-column stores they are considered more read optimized as writing data is organized and indexed, which allows for fast reading [25].

Graph Databases

In graph databases the data is represented as a set of vertices linked together by edges [25]. They are designed to fill the gap between graph data modeling requirements and the tabular abstraction in traditional databases [20] by explicating links among vertices. Most graph databases providers implement "property graphs" [11, 21] in which both nodes and edges may have properties in the form of key-value pairs and a name, and edges are binary and directed.

Graph databases are gaining popularity, mainly because they often provide advanced and expressive query languages [25], such as SPARQL, which is a declarative query specification designed for most of the available graph databases [19]. Furthermore, they were found suitable for finding relationships within huge amounts of data at a very fast rate due to the index free adjacency [19].

As in most of the other NoSQL databases, integrity constraints are poorly studied and implemented in graph databases [2]. Another disadvantage is that partitioning the data is extremely hard and complicated due to the direct access to heavily interconnected data [21]. This problem is unique to the graph databases in the NoSQL era, since graph databases are the only NoSQL type that emphasizes the relations between the data.

2.4 NewSQL Databases

NewSQL is a term coined to describe a new group of databases that share much of the functionality of traditional SQL relational databases, while offering some of the benefits of NoSQL technologies [33]. NewSQL databases use the relational data model and SQL and try to address some of the relational databases disadvantages that were addressed by NoSQL. They support ACID [33] and therefore offer stronger consistency than in most of the NoSQL systems ("Eventual Consistency").

Since NewSQL support ACID, they are good for transactions driven applications, but they are also scalable and distributed, which makes them work better than the relational model when there is a heavy workload. NewSQL databases are a hybrid solution which provides a scalable performance for OLTP workloads [33], which mostly did not exist in the relational (workload) or NoSQL (transactions) databases. In particular NewSQL aims at (1) read-write transactions that are short-lived; (2) read-write transactions that touch a small subset of data using index lookups; and (3) read-write transactions that are repetitive [29]. A limitation of the NewSQL database is the support for different kind of data [33].

Table 1. Database comparison

Database	Property				
	Consistency	Integrity	Performances	Query options	Flexibility
Relational	Immediate consistency	Yes	Scale up	SQL	No
OO	Immediate consistency	No	Scale up	OQL	No
NoSQL-KV	Eventual consistency	No	Scale out	No, though some exist (e.g., Redis)	Yes
NoSQL-D	Per Op.	No	Scale out	No, though some exist (e.g., Mongo)	Yes
NoSQL-C	Per Op.	No	Scale out	No, though some exist (e.g., CQL in Cassandra)	Yes
NoSQL-G	Mostly immediate	No	Scale out	No, though some exist (e.g., Cypher in Neo4j)	Yes
NewSQL	Mostly immediate	Yes	Scale out	SQL	No

2.5 Summary

Table 1 summarizes the support for the various properties defined before for each database. Relational, Object-oriented (OO), NoSQL Key-Value (KV), Document bases (D), Column Oriented (C), Graph (G) and New SQL. In NoSQL databases some functionalities are defined per each operation (read or write). This concept represented in the table by the abbreviation "Per Op". Scale out refers to horizontal scaling, whereas scale up refers to vertical scaling.

As Table 1 shows, immediate consistency is supported in centralized database management, integrity is maintained only in relational databases, the new technologies support scaling out the databases, query languages and their standardization are supported mainly in mature databases, and flexibility is supported by the new database technologies.

3 Database Design

The traditional database design process has three steps [4]: (1) **Conceptual modeling:** results in a conceptual model of the database that ignores the logical view and the chosen technology; (2) **Logical modeling:** receives a conceptual model and yields a database schema; and (3) **Physical modeling:** receives a database schema and yields a concrete implementation of the schema. The three steps are mainly used in the relational database design, yet they have proved their benefits as they are easy to learn and achieve good results [13].

It is important to design the right database solution based on goals, technologies and what is required by the application. The surveyed papers and approaches lead to the definition of the desired characteristics from a database design approach:

- **Expressiveness**
  - **Data Requirements** refer to the method support for addressing real-world concepts such as types of entities and their relationships such as inheritance and aggregation [e.g., 7, 8].
  - **Non Functional Requirements (NFR)** refer to the ability to express meta-properties such as data volume, variety, velocity and veracity [e.g., 1].
  - **Functional Requirements** refers to the queries, i.e., the questions that need to be answered by the applications. A design might differ based on the different goals of the application. Goals include queries, their type, and their frequency [e.g., 8, 27].
- **Ease of Use**
  - **Guidelines** of how to use the method should be given and explained thoroughly. A user should be able to create a correct and complete database easily. Such guidelines would increase the usability of the method [e.g., 1, 27].
- **Formality** is a key issue of a design approach as it facilitated automation and reasoning. Such formality would allow for correctness checking, select the most suitable database solution, perform what-if analysis, and a like [e.g., 7, 27].

In light of these characteristics in the following we review database design approaches.

3.1 Design Approaches for Relational Databases

The traditional database design process which fits the relational databases flawlessly is well researched and mature [4]. The process is known to be important in the relational world since it has to take into account storage space and data redundancy [8].

The first design step of conceptual modeling has many options, the most used are ERD and Class diagram, in which entities and relations are defined. In the logical phase there is a schema that contains suggested tables and keys. In order for the logical schema to be correct and efficient there is a need to normalize the schema [12, 37]. Normal forms disallow null values, repetitive values and such. Based on the final logical schema, a set of tables can be derived and set the physical design. In the physical design some tables and/or views may be added based on needs (such as aggregate tables). It is important to mention that queries are rarely used as part of the first design stage. Query considerations usually result in further indexes and aggregate tables/views.

3.2 Design Approaches for Object-Oriented Databases

In object-oriented databases most design is done by preconceived notions of the designers and programmers, and share characteristics with programming object-oriented applications with common terms such as objects, encapsulation, inheritance, etc. In object-oriented databases the database specification comprises class definition in the programming languages[7]. In order to create an object-oriented database, the key steps are: identifying classes, objects, the relations between them such as inheritance or aggregation and define their behavior [18]. This can be done in object-oriented design using UML class diagram[8]. For example, Ou [28] uses the class diagram as a basis for object-oriented databases. His research adds the ability of adding integrity constraints to the class diagram in order for the database to maintain data integrity.

3.3 Design Approaches for NoSQL Databases

Currently, there is no standard or widely used ways to design NoSQL databases. Nevertheless, it is recognized that “NoSQL systems require standard query languages and discipline for modeling” [33]. It is also stated that “data modeling has an impact on querying performance, consistency, usability, software debugging and maintainability, and many other aspects” [15]. As a possible solution, it is argued that big data requires careful modeling, which can be done by conceptual modeling by trained developers [13]. In [15] they claim that big data requires data modeling and support this claim by

[7] http://www.odbms.org/wp-content/uploads/2013/11/Tutorial.Cattell.2009.pdf.

[8] https://stackoverflow.com/questions/1100819/how-do-you-design-object-oriented-projects.

an experiment that presents different data models and their different performance results, both in run-time and in code evaluation.

Many of the methods that we review in this section use known conceptual modeling techniques, such as ERD or UML class diagram. NoSQL might be schema less, but entity-relationship diagrams are still useful for designing a NoSQL schema [33].

Kaur and Rani [19] review the different data models and query options and demonstrate several use-cases of data modeling. Nevertheless, the paper lacks in introducing rules or guidelines for designing NoSQL databases, although the authors do imply that the type of queries has to be kept in mind while designing such databases.

Bugiotti et al. present a design methodology for NoSQL databases, excluding graph databases [7]. They use the fact that some initial activities are independent of the specific target system and that design can be made without any prior knowledge regarding the specific NoSQL solution. Their approach (NoAM) exploits the observations that various NoSQL systems share similar modeling features. It introduces some new concepts such as Blocks, Entries, and Aggregates. Their approach as many others is based on a conceptual model, a UML class diagram. This method provides a comprehensive solution and provides good fundamentals for future expansions. The method treats different NoSQL databases with the same modeling concepts; however, it fails in analyzing the tradeoffs among these. The authors explain well the new concepts but put less emphasis on the process and its guidelines. Their evaluation presents how different design concepts that were suggested in the paper impact different queries.

Design Approaches for Key-Value Databases

Research related to the design of key-value databases has not gained much of attention, probably due to their simplicity. Nevertheless, Rossel and Manna suggest a modeling methodology for the key-value databases [30]. Its starting point is a conceptual model, which can be either in ERD or UML class diagram, and access patterns. The methodology proposes that the logical design is represented by a set of interrelated datasets, defined as a 4-tuple: DN-dataset name (usually it corresponds to an entity in the conceptual model), NS-Namespace, KS-The set of key parts (helps assemble complete key to access the value by concatenating them) and V-the value. Relationships are represented as a tuple where the first element is a tuple of two datasets, and the second element is a list of links, i.e., how to obtain from a concrete element in the first dataset the information to assemble the key to have access to a concrete element in the second dataset. The method lacks in providing rules and guidelines for transforming conceptual models into logical ones. In addition, no evaluation is reported.

Design Approaches for Wide-Column Stores

Chebotko et al. suggest a modeling methodology for Apache Cassandra [8]. It is query driven and is based on a conceptual model and an application workflow. The application workflow describes access patterns or queries that a data-driven application needs to run against the database. Each access pattern specifies what attributes to search for, search on, order by, or do aggregation on with a distributed counter. The conceptual modeling is done with an ERD ("because this notation is truly technology-independent and not tainted with any relational model features"), but can be done with any conceptual modeling method. They also define five mapping rules to transform the conceptual data model to a logical one, based on the properties and capabilities of

Cassandra. The authors implemented a web-based tool called KDM that automates the suggested method. Though a well-defined methodology is presented, it is tailored to a specific database. To overcome this specificity, Mior et al. propose a model that fits all the column stores [27]. It is a tool that recommends the needed column families in the application. The tool is called NoSE (NoSQL Schema Evaluator) and receives as inputs a conceptual model (an entity graph that is a restricted type of an ERD) and a description of the expected workload. The outputs are the recommended schema, which describes the column families that should be used to store the applications' data and a set of plans, one plan for each query and update in the workload. The target applications' workload is described as a set of parameterized query and update statements. Each query and update is associated with a weight indicating its relative frequency in the anticipated workload. The tool is comprehensive and uses only known or easy-to-learn elements. It also takes into account a lot of variables that ensure that a good schema will be chosen. The approach has also been evaluated.

Boussahoua et al. try to answer the question of how to organize data in column families to serve effectively specific queries, in particular, in the case the OLAP queries (in which the read load exceeds the write load), as in a data warehouse [6]. Their strategy (CN-DW) has 4 steps: (1) Extracting the set of attributes of the fact table and dimension tables; (2) Grouping of attributes and constructing column families by K-means; (3) Generating a logical schema of the CN-DW; (4) Preparing and loading the data into CN-DW. The method uses a set of known tables and workload that consists of frequent queries. Based on attributes used in the queries they build two matrices. The matrices are then used as an input to K-means, which outputs the set of column families needed. The work assumes that a data warehouse in the form of fact and dimension tables already exists. The method is evaluated by comparing different K-means setting to two other modeling options (flat and naïve).

Design Approaches for Document Stores

Vera et al. propose a standard for NoSQL data modeling [36]. This proposal uses NoSQL document-oriented databases aiming to introduce modeling techniques that can be used on databases with document features. It addresses the type of relationships between two documents either embedded or referenced. The paper proposes a "database viewing" of how to model documents and their relationships in a clear and visual manner. Nevertheless, no rules or guidelines are specified.

Mason focuses mainly on data modeling design techniques for a MongoDB database [26]. He uses an ERD as the logical data model (LDM). The main focus is how to convert the LDM to an appropriate document-oriented physical data model (PDM). As in the previous paper the work focuses on the relationships between documents - embedded or referenced. However, it suggests a way to choose a relationship type for two known documents. The relationships are chosen based on five heuristics that take into account the data itself and needed queries. The method provides a good starting point by setting some rules that should be addressed. However, the ERD lacks constructs such as hierarchy and aggregation. Furthermore, the notion of query frequency is not addressed.

Shin et al. propose designing NoSQL databases based on a conceptual model in the form of a UML class diagram [32]. The work focuses on the need in the design phase for NoSQL databases. The method is rather simple and easy to use, but does not show a way of distinction between relationship types, which may impact queries performance.

de Lima et al. propose an approach for designing document databases based on a conceptual schema and a workload information [10]. The transformation of the conceptual schema to the document logical schema is based on a set of rules applied based on the workload information. Their conceptual model of choice is an Extended Entity Relationship. The method was evaluated by comparing the full method to the method without the workload information. The method is quite comprehensive and inclusive of all ER model constructs.

Design Approaches for Graph Databases

De Virgilio et al. propose a modeling method based on ERD [11]. They claim that the main goal in the design of a graph database is the minimization of data access operations needed in graph traversals at query time. The first phase is generating an Oriented ER (O-ER) diagram from the regular ERD, which is basically a directed labeled and weighted graph. In the second phase the O-ER diagram is partitioned based on another set of rules. The third phase generates a template for the final database. The template is kind of a schema but less rigid. It describes the data types in a graph and the way they are connected. The database instance is not forced to conform to the template in a rigid way. Rather, it is the initial structure of the graph database that can be later extended or refined. The proposed method was evaluated and compared to databases obtained with a "Sparse" strategy, in which the properties of an object with n properties is decomposed into a set of N different nodes. The databases were compared based on response time. The method is easy to understand and use, yet it can benefit from further improvements with respect to its expressiveness (e.g., inheritance and aggregation) and its evaluation as the "Sparse" strategy generates many nodes and relationships, which impacts the response time.

Daniel et al. propose a framework that translates conceptual schemas expressed in UML into a graph representation [9]. UMLtoGraphDB proposes a structured methodology for systems development that promotes the separation between a specification defined in a platform independent way (Platform Independent Model, PIM), and the refinement of that specification adapted to the technical constraints of the implementation platform (Platform Specific Model, PSM). The UML and the OCL conform to the PIM level. The framework includes rules to transform a class diagram to a graph model (based on a suggested Meta model). The OCL constraints are transformed into queries in Gremlin, a basic graph query language, which is adopted by several graph databases. Once these are done, the two previous steps are generated into code that will be used to access the physical graph database. The method and the tool are very well written and comprehensive. However, the method may benefit from further refinements of some rules. For example, each association class turns into a vertex definition, while maybe it is better to add it as properties on the relation. This would probably be better addressed if the method addresses query considerations.

Akoka et al. [1] suggest a method that takes into account the big data four Vs: variety, velocity, volume, and veracity. Their conceptual model of choice is an ERD. For each component in the (ERD) meta model, they added information regarding the four Vs. They use two set of rules. The first is to allow the translation of the ERD into a logical property graph model and the second is to allow for the transformation of the logical model to a script. The resulting graph contains data with fictitious values in realistic size. The Volume is the major property of the method. It is used to create the exact number of (fictitious) nodes. The other V's, are less discussed and seems to add less to the method when creating, as opposed to evaluating a database. The four V's s are valuable as part of the NFR but their contribution seems to be limited.

Roy-Hubara et al. proposed a modeling methodology for graph databases based on ERD and a schema definition called GDBS [31]. The authors defined schema notations for the graph databases and defined a process based on two sets of rules transforming a rich ERD to the suggested schema. The schema then defines the constructs that will exist in the graph database in order to maintain data correctness and integrity. The method provides a good start for a design method but lacks in supporting query considerations. Furthermore, no implementation or evaluation were performed.

3.4 Design Approaches for New SQL Databases

By definition, NewSQL databases are "stronger" Relational databases as they are relational, have a predefined schema, support transactions and ACID, and queried in SQL, but as opposed to the older relational databases they scale-out better and work well on a distributed environment[9] [29, 33]. This means that when designing a NewSQL database, it is probably preferable to use the same notions and methods that are done when creating a relational database, since they support the same tabular structure, and same data types.

4 Discussion

As seen in the previous section, there are many methods for designing databases. Some are accepted and learned in academia and some are still being developed and requires validation and adoption evaluation.

Some of the methods indeed address the requirements defined previously. Most of the methods address the data expressiveness requirements with a conceptual model in the form of an ERD or a UML class diagram. The two techniques are vastly used in the relational model and were also adopted by NoSQL studies, probably due to their expressiveness and familiarity. As these models are widely spread, the learning process is easier, a fact that will assist in adopting new design methods.

Several of the methods considered query needs in the design process. Some used query types (OLAP vs. OLTP, insert vs. select), query frequencies, and even specific queries. In order to obtain a good design, it is important to know when and what the data is used for. Queries may affect the design of indexes, relations, and data blocks (i.e., column families, nodes and such).

Most methods refer to NFR considerations only to a limited extent. However, NFRs are most crucial when choosing a database. For example, a relational database may be a

[9] https://www.datasciencecentral.com/profiles/blogs/newsql-rdbms-on-steroids.

wrong solution when it comes to large volume. NoSQL document stores may be less suitable to a database design that relies heavily on data writes.

Another important issue is the need for proper rules and guidelines. Some methods lack specific and detailed guidelines, a fact that may impact their adoptions as missing rules may lead to wrong solutions or discarding of the design method. The guidelines should be complete, i.e., users should be able to use the guidelines to achieve a proper database design.

Gathering the needs and solutions proposed by the various design approaches results in the following features that are required to be taken into consideration while designing databases:

- **Specify the data types and their relationships:** the data needed should be thoroughly stated - what entities are to be in the application, their roles, and interactions. This can be accomplished by conceptual modeling.
- **Estimate the four Vs for each of the above:** The four V's should be examined for each entity in order to choose and design the right solutions.
- **Analyze access pattern:** Type of queries and patterns has to be determined. Read vs. write queries, and more specifically what entities and properties are used. This can be done with parameterized queries or application workflow.
- **Assess workload:** For each access pattern, it is important to assess its effect on the final workload. Queries with higher frequencies should have more effect on the final design. It might be hard to obtain exact numbers at such early stage, but an estimation is possible and required.

With respect to the various methods we observe that detailed guidelines are needed and further evaluations of their usage and benefits are still required. A major concern regarding the methods is that they a-priori enforce the database technology instead of selecting the right technology based on the design and analysis of the application requirements.

5 Conclusions

The new era of databases requires rethinking of design methods. In this paper we review the state-of-the-art database design methods. We define the requirements from these methods and analyze these in light of these requirements. Unlike traditional database design, in which the focus was on the data, the new era of databases design requires further considerations that should be taken at early stages. These include the estimation of the four Vs and the analysis of access patterns, workload and queries. A major issue we identified is that there is no holistic approach that first consider the requirements and then decides upon the needed database technology and the actual model.

In this work we provide an initial exploration of the requirements from a database design approach. In the future, we plan to further examine the needs and look into options of how to address these needs within a comprehensive database design framework. This may include facilities for assisting in the choosing proper databases.

References

1. Akoka, J., Comyn-Wattiau, I., Prat, N.: A four V's design approach of NoSQL graph databases. In: de Cesare, S., Frank, U. (eds.) ER 2017. LNCS, vol. 10651, pp. 58–68. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70625-2_6
2. Angles, R.: A comparison of current graph database models. In: 2012 IEEE 28th International Conference on Data Engineering Workshops (ICDEW), pp. 171–177, April 2012
3. Atzeni, P., Jensen, C.S., Orsi, G., Ram, S., Tanca, L., Torlone, R.: The relational model is dead, SQL is dead, and I don't feel so good myself. ACM SIGMOD Rec. **42**(2), 64–68 (2013)
4. Badia, A., Lemire, D.: A call to arms: revisiting database design. ACM SIGMOD Rec. **40** (3), 61–69 (2011)
5. Berg, K.L., Seymour, T., Goel, R.: History of databases. Int. J. Manag. Inf. Syst. **17**(1), 29 (2013)
6. Boussahoua, M., Boussaid, O., Bentayeb, F.: Logical schema for data warehouse on column-oriented NoSQL databases. In: Benslimane, D., Damiani, E., Grosky, W.I., Hameurlain, A., Sheth, A., Wagner, R.R. (eds.) DEXA 2017. LNCS, vol. 10439, pp. 247–256. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-64471-4_20
7. Bugiotti, F., Cabibbo, L., Atzeni, P., Torlone, R.: Database design for NoSQL systems. In: Yu, E., Dobbie, G., Jarke, M., Purao, S. (eds.) ER 2014. LNCS, vol. 8824, pp. 223–231. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12206-9_18
8. Chebotko, A., Kashlev, A., Lu, S.: A big data modeling methodology for Apache Cassandra. In: IEEE International Congress on Big Data, pp. 238–245, June 2015
9. Daniel, G., Sunyé, G., Cabot, J.: UMLtoGraphDB: mapping conceptual schemas to graph databases. In: Comyn-Wattiau, I., Tanaka, K., Song, I.-Y., Yamamoto, S., Saeki, M. (eds.) ER 2016. LNCS, vol. 9974, pp. 430–444. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46397-1_33
10. de Lima, C., dos Santos Mello, R.: A workload-driven logical design approach for NoSQL document databases. In: Proceedings of the 17th International Conference on Information Integration and Web-Based Applications and Services, p. 73. ACM, December 2015
11. De Virgilio, R., Maccioni, A., Torlone, R.: Model-driven design of graph databases. In: Yu, E., Dobbie, G., Jarke, M., Purao, S. (eds.) ER 2014. LNCS, vol. 8824, pp. 172–185. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12206-9_14
12. Design, S.: Relational Database Design (1995)
13. Embley, D.W., Liddle, S.W.: Big Data—conceptual modeling to the rescue. In: Ng, W., Storey, Veda C., Trujillo, J.C. (eds.) ER 2013. LNCS, vol. 8217, pp. 1–8. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41924-9_1
14. Gessert, F., Wingerath, W., Friedrich, S., Ritter, N.: NoSQL database systems: a survey and decision guidance. Comput. Sci. Res. Dev. **32**(3–4), 353–365 (2017)
15. Gómez, P., Casallas, R., Roncancio, C.: Data schema does matter, even in NoSQL systems! In: 2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS), pp. 1–6, June 2016
16. Han, J., Haihong, E., Le, G., Du, J.: Survey on NoSQL database. In: 2011 6th International Conference on Pervasive Computing and Applications (ICPCA), pp. 363–366. IEEE, October 2011
17. Haseeb, A., Pattun, G.: A review on NoSQL: applications and challenges. Int. J. Adv. Res. Comput. Sci. **8**(1), 203–207 (2017)

18. Holland, I.M., Lieberherr, K.J.: Object-oriented design. ACM Comput. Surv. (CSUR) **28**(1), 273–275 (1996)
19. Kaur, K., Rani, R.: Modeling and querying data in NoSQL databases. In: IEEE International Conference on Big Data, pp. 1–7, October 2013
20. Khazaei, H., Fokaefs, M., Zareian, S., Beigi-Mohammadi, N., Ramprasad, B., Shtern, M., Litoiu, M.: How do I choose the right NoSQL solution? A comprehensive theoretical and experimental survey. Big Data Inf. Anal. (BDIA) **2**, 1 (2016)
21. Kolomičenko, V., Svoboda, M., Mlýnková, I.H.: Experimental comparison of graph databases. In: Proceedings of International Conference on Information Integration and Web-Based Applications and Services, p. 115. ACM, December 2013
22. Lake, P., Crowther, P.: Concise Guide to Databases. Undergraduate Topics in Computer Science. Springer, London (2013). https://doi.org/10.1007/978-1-4471-5601-7
23. Leavitt, N.: Whatever happened to object-oriented databases? Computer **8**, 16–19 (2000)
24. Leavitt, N.: Will NoSQL databases live up to their promise? Computer **43**(2), 12–14 (2010)
25. Lourenço, J.R., Cabral, B., Carreiro, P., Vieira, M., Bernardino, J.: Choosing the right NoSQL database for the job: a quality attribute evaluation. J. Big Data **2**(1), 18 (2015)
26. Mason, R.T.: NoSQL databases and data modeling techniques for a document-oriented NoSQL database'. In: Proceedings of Informing Science and IT Education Conference (InSITE), pp. 259–268 (2015)
27. Mior, M.J., Salem, K., Aboulnaga, A., Liu, R.: NoSE: schema design for NoSQL applications. IEEE Trans. Knowl. Data Eng. **29**(10), 2275–2289 (2017)
28. Ou, Y.: On using UML class diagrams for object-oriented database design. In: Bézivin, J., Muller, P.-A. (eds.) UML 1998. LNCS, vol. 1618, pp. 173–188. Springer, Heidelberg (1999). https://doi.org/10.1007/978-3-540-48480-6_14
29. Pavlo, A., Aslett, M.: What's really new with NewSQL? ACM SIGMOD Rec. **45**(2), 45–55 (2016)
30. Rossel, G., Manna, A.: A modeling methodology for NoSQL key-value databases. Database Syst. J. **8**(2), 12–18 (2017)
31. Roy-Hubara, N., Rokach, L., Shapira, B., Shoval, P.: Modeling graph database schema. IT Prof. **6**, 34–43 (2017)
32. Shin, K., Hwang, C., Jung, H.: NoSQL database design using UML conceptual data model based on Peter Chen's framework. Int. J. Appl. Eng. Res. **12**(5), 632–636 (2017)
33. Storey, V.C., Song, I.Y.: Big data technologies and Management: what conceptual modeling can do. Data Knowl. Eng. **108**, 50–67 (2017)
34. Tang, E., Fan, Y.: Performance comparison between five NoSQL databases. In: 2016 7th International Conference on Cloud Computing and Big Data (CCBD), pp. 105–109, November 2016
35. Tudorica, B.G., Bucur, C.: A comparison between several NoSQL databases with comments and notes. In: 2011 10th Roedunet International Conference (RoEduNet), pp. 1–5, June 2011
36. Vera, H., Wagner Boaventura, M.H., Guimaraes, V., Hondo, F.: Data modeling for NoSQL document-oriented databases. In: CEUR Workshop Proceedings, vol. 1478, pp. 129–135 (2015)
37. Wesley, D.: Relational database design. J. Insur. Med. NY **32**(2), 63–70 (2000)

Evaluation of a Design Method for Graph Database

Noa Roy-Hubara[1](✉), Lior Rokach[1], Bracha Shapira[1], and Peretz Shoval[1,2]

[1] Ben-Gurion University of the Negev, Be'er Sheva, Israel
nro@post.bgu.ac.il
[2] Netanya College, Netanya, Israel

Abstract. We present an evaluation of a new design method of a graph database schema (GDBS). The method is based on an Entity-Relationship diagram (ERD) of the domain of application, which is mapped to a GDBS in a two-step process, according to specific rules. Following the development of the method, we evaluate it via a controlled experiment in which we compared it with alternative design methods. The results indicate that the proposed method improves the design of a graph database.

Keywords: Graph database · Conceptual modeling
Conceptual schema · ERD · GDBS · NoSQL

1 Introduction

The recent decade had brought about massive growth in the Web and Web data - pictures, documents, videos and many more. What worked well for years with relational databases is not well suited for the unstructured massive amounts of data and applications that are part of the Web, such as social networks. As a result, new types of database have emerged, including the NoSQL ("not only SQL") databases. Graph databases are one of the NoSQL database types.

Nowadays, there are only a few design methods for graph databases [1, 4]. Current approaches lack formal guidelines (i.e. full process description) and important data components. To address existing limitations we proposed a design method of a database schema for graph databases [7].

The method for modeling a graph databases schema (GDBS) is based on an entity-relationship diagram (ERD) of the domain of application, which is mapped to a graph database in two steps: in the first step, the original ERD, which includes constructs (such a ternary-relationships) that cannot be mapped directly to a graph database schema, is mapped to an equivalent ERD with alternative constructs that can be mapped to a graph database; in the second step, that ERD is mapped to the target graph database according to certain mapping rules. The resulting graph database is expressed in form of a diagram and as sort of DDL statements that can be added to a future definition language of a graph database management system.

In this study we describe an evaluation of that method. In the evaluation we conduct an experiment in which we compare three alternative design methods, and considering three aspects: quality of the resulting database, time taken to design, and designers' satisfaction from the methods.

The rest of this paper is structured as follows. In Sect. 2 we survey the related works on graph databases design. In Sect. 3 we describe briefly the graph database model. Section 4 describes our method for modeling the GDBS, and in Sect. 5 we describe the experiment in which we compared three alternative methods for designing graph database. Section 6 summarizes and set plans for future research.

2 Related Work

As stated before, the design of graph database relies on best practices. Nevertheless, a few design methods for graph databases were developed. In this section we review these methods,

De Virgilio, Maccioni and Torlone propose a modeling method based on an ERD [4]. The first phase in their method is generating an Oriented ER (O-ER) diagram from the regular ERD - a directed, labeled and weighted graph. In the second phase the O-ER is partitioned based on a set of rules. The third phase generates a template for the final database. The template is kind of a schema that describes the data types in a graph and the way they are connected. The database instance is not forced to conform the template in a rigid way. Rather, it is the initial structure of the graph database that can be later extended or refined. The method is easy to understand and use, yet, it can benefits from further improvements with more ERD constructs (e.g., inheritance and aggregation).

Daniel, Sunyé and Cabot propose a framework that translates conceptual schemas expressed in UML into a graph representation [3]. UMLtoGraphDB proposes a structured methodology that promotes the separation between a specification defined in a platform independent way (Platform Independent Model, PIM), and the refinement of that specification adapted to the technical constraints of the implementation platform (Platform Specific Model, PSM). The UML and the OCL conform to the PIM level. The framework includes rules to transform a class diagram to a graph model. The OCL constraints are transformed into queries in Gremlin, a basic graph query language, which is adopted by several graph databases. Finally, the two previous steps are generated into code that will be used to access the physical graph database. The method and the tool are very well written and comprehensive. However, the method lacks in some rules. For example, each association class turns into a vertex definition, while maybe it is better to add it as properties on the relation. This will probably be better answered if the methods would address query considerations.

Akoka, Comyn-Wattiau and Prat [1] suggest a method that takes into account the big data four Vs: variety, velocity, volume and veracity. The method uses a conceptual model - an ERD. As opposed to some previous papers, the authors address n-ary relations and "is-a" links. To each component in the model, they added information regarding the four Vs. They use two set of rules: the first is to allow the translation of the ERD into a logical property graph model, and the second is to allow to transformation the logical model to a script. The resulting graph contains data with fictitious values in realistic size. While the Volume is the major property of the method, the other V's are less discussed and seems to add less to the method when creating, as opposed to evaluating a database.

Another observation that we had is that although the methods provide the basis for designing graph databases; they were evaluated only to a limited extent.

3 The Graph Database Model

In this section we provide a brief description of the graph database model we use in this study. For clarity, we use an example from the domain of movie recommendation system, which provides information about movies, their actors, directors, and also stores users' ratings of movies (The details of the requirements of this example can be found in [7]). Figure 1 presents the ERD of the example. For clarity, here are some explanations of the ERD:

- Actor, Director, Producer and User are sub-types the Person. The sub-types are not exclusive; an actor may also be a director; a producer may also be an actor, etc. The sub-types cover all possible persons in the system, i.e., there are no other sub-types of Person. This is indicated by the "T" connecting the sub-type arrows.
- Rate is a ternary relation (many-to-many-to many) between User, Movie and Rating: a movie may be rated by many users with different ratings. Each rating has a date, and actually, date is a partial key of the ternary relationship.

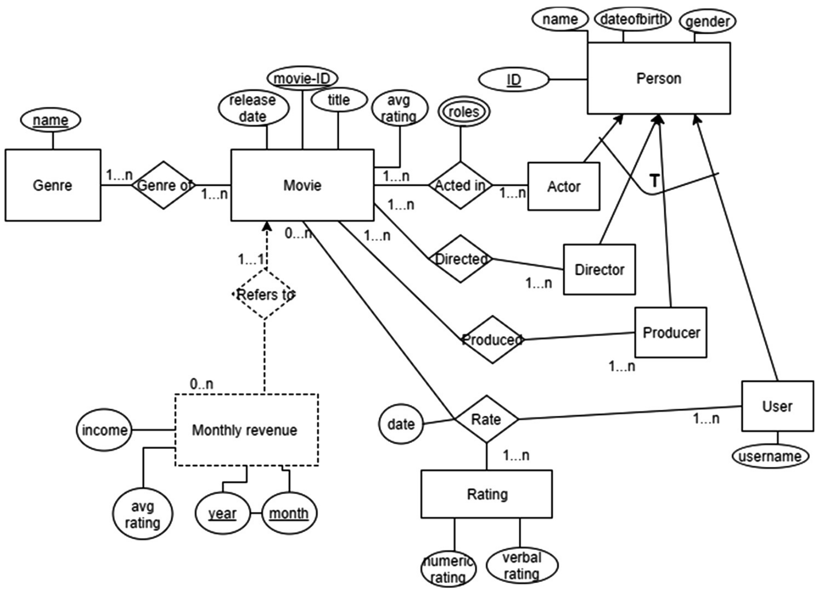


Fig. 1. ERD of the movies recommendation system

- Monthly revenue is a weak entity related to its "strong" entity Movie; it stores monthly information for each movie. Note its partial key attributes year and month.

3.1 Components of a Graph Database Model

The graph database model used in this study consists of nodes, edges, and properties, as in [6], with some extensions. The graphic notations of the components of this model are presented in Fig. 2.

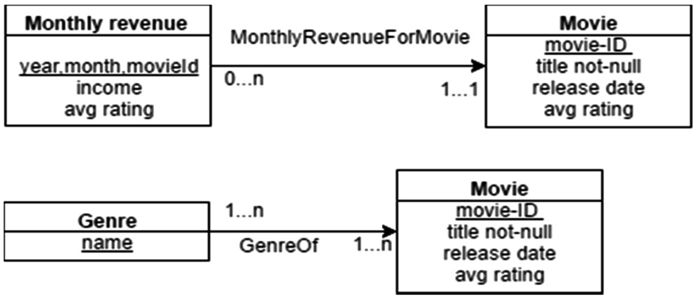


Fig. 2. Examples of nodes, edges, properties, and cardinality constraints

Node: A node represents an entity in the real world. A node has a label (its name) and properties, including is a key property which enables it unique identification. In Fig. 2 Movie is a node which is uniquely identified by the movie-id property.

Edge: An edge represents a binary relationship between two nodes. As defined in most graph databases [6], an edge has a direction, meaning that it has a start node and an end node. An edge may also have properties and has a label. In Fig. 2 the edge GenreOf has start node, Genre and the end node Movie.

Property: As said, both nodes and edges may have properties. Properties may have constraints. Possible constraints include:

- Key: As said, each node has a key which may be one property or a combination of several properties. For example, in Fig. 2, movie-ID is the key of Movie and the combination of movie-ID, year and month is the key of Monthly Revenue.
- Not-Null: The property must have a value (i.e., exist) for all instances of the node or edge. For example, in Fig. 2, the title is a Not-Null property of Movie (assuming that every movie must have a title).
- Set: The property may have many values. For example, 'roles' is a set property of the edge between Movie and Actor, since an actor may play many roles (this does not in Fig. 2.).

Cardinality constraints

The graph database model we use extends existing graph database models by including cardinality constraints between the nodes of each edge. For this we use the same notations used in the ERD (i.e., next to each node we write the min. and max. number of nodes that may participate in each edge type).

For example, Fig. 2 shows a one-to-many relationship between Movie and Monthly revenue, and a many-to-many relationship between Movie and Genre.

4 The Method for Modeling the Graph Database Schema

Our method for modeling the graph database schema (GDBS) is based on an ERD of the domain of application which is mapped to the GDBS in two steps [7]: (a) adjusting the original ERD to an equivalent ERD that is ready for mapping to a GDBS; and (b) mapping the adjusted ERD to the GDBS.

4.1 Adjusting the Original ERD

An ERD may include constructs which cannot be mapped directly to a GDBS, which consists of only nodes and binary edges. Therefore, in the first step we adjust some of the original ERD constructs to an equivalent ERD constructs that can be mapped later on, as follows.

Ternary relationships
A ternary relationship is mapped to a weak entity, with binary relations to the entities involved in the ternary relation. If the relation has properties, they are added to the weak entity. The name of the weak entity may be identical to the name of the original ternary relationship, or be any name that resembles its role. For example, the ternary relation between the entities User, Movie, and Rating is mapped to a weak entity with three binary relationships to the above three entities, as shown in Fig. 3.

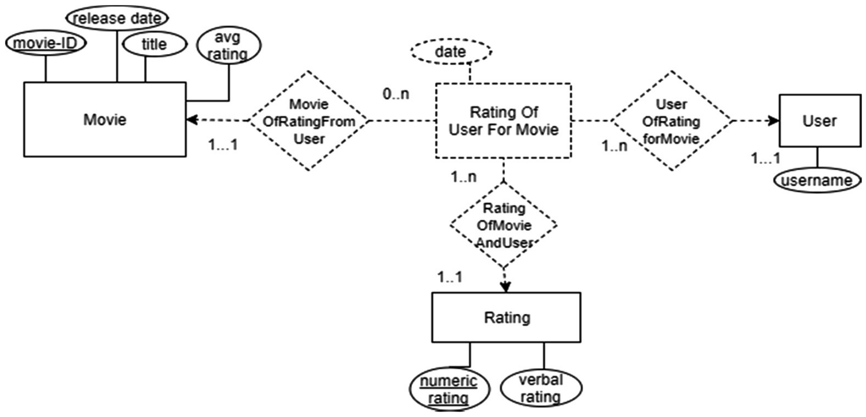


Fig. 3. Ternary relation to 3 binaries relations

Aggregation (whole-parts) relationships
An aggregation relationship is mapped to an "ordinary" binary relationship, in which the cardinality of the parts entity is 0:n or 1:n (depending whether the parts entity is mandatory or not), while the cardinality of the whole entity is always 1:1.

Inheritance (is-a) relationships
Inheritance relationships can be mapped in two different ways.
A: no change of the original entities: According to this method, the involved entities remain unchanged, but the "is-a" relationships are mapped to "ordinary" binary relationships with cardinalities 1:1 next to each of the involved entities.

B: removing the inheritance relationships: Based on this method, we remove the inheritance relationships, and merge the super-entity with the sub-entities. We distinguish between two cases:

1. **Removing the sub-entities and moving their attributes and relationships up to the super-entity**: This mapping is applied when the inheritance relationship is not defined with the "T" (Total cover) constraint and/or not defined with "X" (Exclusive), meaning that there are super-entities which are not one of the sub-entities and/or that a super-entity may belong to many sub-entities. For example, Person is super-entity of four sub-entities; assuming that there is no "T" and no "X" constraints, the four sub-entities are removed, and their properties and relationships are added to Person
2. **Removing the super-entity and moving its properties and relationships to each of its sub-entities**: This mapping is applied when there are "T" and "X" constraints between the sub-entities, meaning that all super-entities belong to one sub-entity only. Therefore, there is no need to maintain the super-entity. For example, the inheritance relationship between Person and its four sub-entities has a T and X constraints;

In order to determine which way is preferred, we performed an experiment that compared two databases that implemented following the two different methods. With respect to response time and query complexity we found that the second option (method B) outperformed the first one (method A). Therefore, we recommend using it, though we do not reject the use of the first one.

4.2 Mapping the Adjusted ERD to the GDBS

The mapping process consists of the following steps:

Mapping entities to nodes
Each entity is mapped to a node; the entity's properties become the node's properties. A weak entity is mapped to a node just like an "ordinary" entity and the key property of this node is composed of the keys of the related "strong" entities of the weak entity, plus the partial key of the weak entity (if this exists).

Mapping relationships to edges
Each relationship between entities is mapped to an edge connecting two respective nodes: a start node and an end node. The edge's name may be the name of the

relationship. It doesn't matter which of the two nodes is defined as the start node and which is defined as the end node, because as said, the graph database enables traversing from node to node in any direction [6]. Thus, the selection may be arbitrary.

Mapping cardinality constraints
To the best of our knowledge, current graph databases do not define cardinality constraints, i.e., min and max number of nodes that may participate in an edge. For example, Neo4j [2], a leading graph DB system, has not (yet) defined such constraints. Thus, such mapping is planned for the future, though the proposed graph database model (appear is Sect. 3) supports this mapping.

4.3 The Resulting GDBS

The resulting GDBS can be presented in form of a diagram and as DDL statements that can be added to a graph database system. Figure 4 presents the model obtained after applying the mapping option of removing the inheritance relationships.

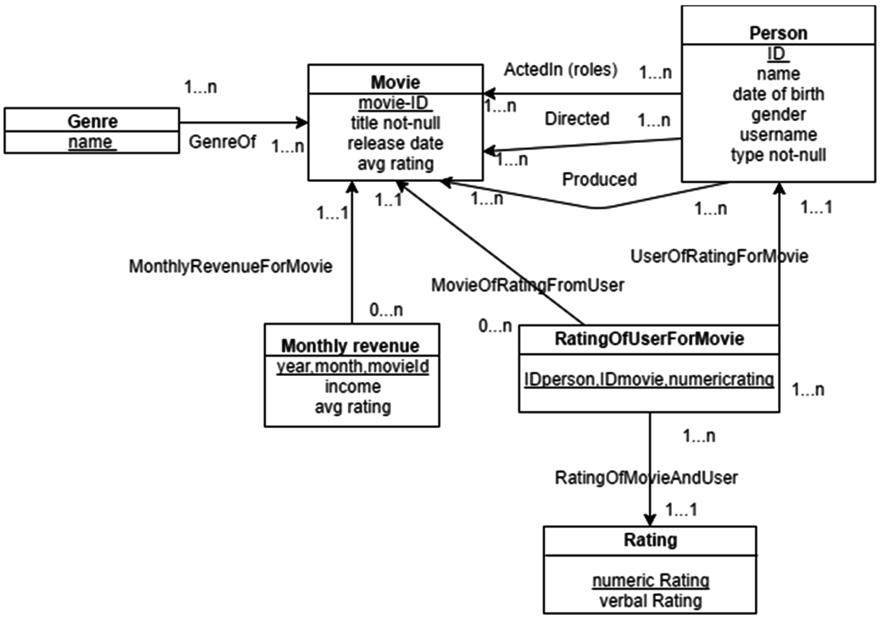


Fig. 4. GDBS without sub-entities and inheritance relationships

5 Evaluating GDBS Design

To evaluate the proposed approach, we set up an experiment that compares it versus "best practice" design, as there is no accepted, structured method for graph database design. Such an evaluation would allow us to test the benefits of the approach and further improve it. In the following we discuss the experiment design, execution, and results.

5.1 Experiment Design

To evaluate the proposed approach we were interested in examining the correctness of the designed DB schema, the efforts required to design, and the usability of the approach. To do so, we defined three dependent variables: (a) correctness of the GDBS created by designers (i.e. how well does the created database schema fits the uses' requirements); (b) time taken to design the GDBS – represents the required efforts; c) designers' satisfaction from the modeling methods – represents the method usability.

The independent variable was the design method. We compared three alternative design methods of a GDBS: (a) "ad-hoc" method, where designers create the model according to "best practices", i.e., they use only a requirements document for a graph database application but are not given any additional information and instructions; this method is assumed to simulate the way designers/programmers create graph databases; (b) a method where the designers are given a requirements document, an ERD, and mapping rules, as we have proposed above; (c) a method where the designers are given a requirements document and an ERD, but without the mapping rules.

The controlled variables were the tasks and the subjects. The same task of designing a GDBS of a certain application was given to all subjects using the three methods. The subjects who performed the task were students of Information Systems Engineering having similar background in databases and information systems development. They were randomly assigned to the above three methods, where each subject used one method. Figure 5 presents the experiment design.

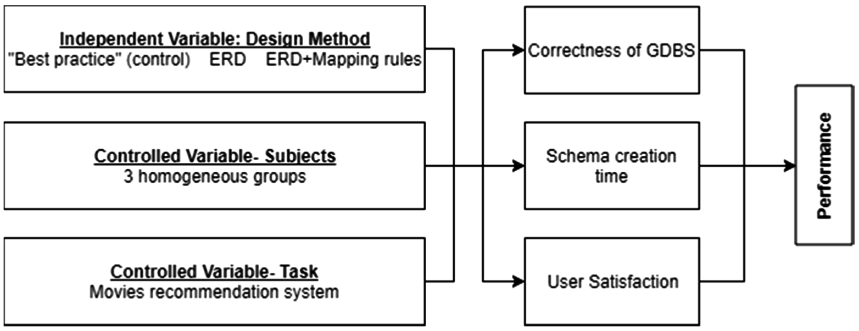

Independent Variable: Design Method
"Best practice" (control) ERD ERD+Mapping rules
Controlled Variable- Subjects
3 homogeneous groups
Controlled Variable- Task
Movies recommendation system
Correctness of GDBS
Schema creation time
User Satisfaction
Performance

Fig. 5. The experimental design

Our conjecture is that the proposed approach would allow the subjects to increase the correctness of the GDBS as it provides guidelines on issues that need to be considered. However, the time to design the database would increase as the subjects have more information to read and instructions to follow. We expect that the satisfaction of the subjects would lean towards the proposed approach, as it is more structured and guided. Thus, we set the following three null hypotheses to be tested:

1. There is no difference in the correctness of GDBS created.
2. There is no difference in the time it takes to complete the task.
3. There is no difference in the satisfaction of methods by the designers.

5.2 Experiment Execution

Twenty subjects participated in the experiment; 12 were 3rd year B.Sc. students of Information Systems Engineering at Ben-Gurion University of the Negev (to who we refer as “regular designers”), and 8 were M.Sc. students of the same department (to who we refer as “experienced designers”). All subjects took a course on databases and two courses on information systems modeling (in which they studied ERD, UML and other modeling methods and languages). None of the subjects had prior knowledge of graph databases; prior to the experiment they were given a lecture on this subject, including explanation of the graph database components, as described above.

The twenty subjects were assigned randomly to the three groups: 8 subjects to the control group, who used the “ad-hoc” method; 6 - to group #1 who used an ERD and the mapping guidelines; and 6 - to group #2 who used an ERD but without the mapping guidelines. The control group included 3 experienced designers and 5 regular designers, group #1 included 2 experienced designers and 4 regular designers, and group #2 included 3 experienced designers and 3 regular designers.

All subjects received the following materials: (a) a requirements document for a graph database of a movie recommendation system; (b) a set of queries that are relevant to this database of movies; (c) a definition of the components of the GDBS; (d) some data needed for the final graph, i.e., data about two movies (so the subjects gain understanding of the actual data). As said, group #1 received, in addition to the above, an ERD of that system (identical to the one presented in Fig. 1) and the mapping rules as defined in Sect. 4. Group #2 received the same materials as group #1, but without the mapping rules. The designers of the experiment, who are experts in the ER modeling, tested the compatibility of the ER model with the requirements document. All subjects were required to produce the same output: a graph database that contains all the data they received as part of the instructions. The final output of each subject was a mini graph database, on paper, that contained the required data.

The start and end times were recorded. At the end of the task, each subject was asked to complete a questionnaire of satisfaction, consisting of two questions, using a scale of 1–5:

1. How easy is it to design a GDBS with the method you used? (1 = very hard; 5 = very easy)
2. How do you assess the quality of your solution? (1 = very bad; 5 = very good)

5.3 Experiment Results

Correctness

Correctness was measured by analyzing the solutions provided by the subjects. Each solution was checked thoroughly for various issues, i.e., problems that impacted the quality and correctness of the solution, compared to the “gold solution” we have prepared in advance. The issues that we came across during the analysis are presented in Table 1. (Some issues were prepared in advanced, while some were first seen after evaluating the subjects’ solutions).

Table 1. Issues (errors) per experimental groups

Issue (Errors)	# of subjects in group #1	# of subjects in group #2	# of subjects in control group
Different nodes for actor, user, etc.	N/a	3	N/a
Two way edges - two opposite edges between the same two nodes	N/a	1	N/a
Rate as an attribute on edge, as opposed to an entity	N/a	4	7
No date saved for rating	N/a	1	1
Edges merge (one edge for directed/produced), added type to Person node	N/a	1	N/a
Staff type nodes	N/a	N/a	3
Attributes as nodes (i.e., SEX was created as a node and not as an attribute)	N/a	1	N/a
Nodes as attributes (i.e., genres created as attributes of movie nodes - not as nodes)	N/a	N/a	1
Edge between rating and monthly revenue (not specified in the requirements)	N/a	N/a	1
Violated database - impossible implementation in the graph databases	N/a	1	1

The table shows the number of subjects within each of the groups who made errors according to the listed issues. Note that the subjects in group #1, who followed precisely the proposed method (i.e., the mapping rules), reached the correct solution and no problematic issues were detected. (One subject only in this group did not follow the instructions and his solution was unclear).

Assuming that all issues are of the same severity, the total score (sum of the number of issues within each group) is 12 for group #2, and 14 for the control group. Each subject in group #2 had on average 2 issues in the solution, while each subject in the control group had on average 1.75 issues in the solution. If we ignore the first issue, which is the most minor and does not prevent a viable solution, group #2 had on average 1.5 issues in the solution.

Based on these results, we conclude that the correctness of the solutions provided by subjects of the group #1, who utilized the modeling methods with both the ERD and the mapping rules, outperformed the other groups. Even group #2 who used an ERD without the guidelines outperformed the control group who used the "ad-hoc" method. These results are in line with our conjecture that the proposed method increases the correctness of a resulting database.

Time

We summed and averaged the time to complete the task within each group and performed ANOVA analysis. Table 2 presents the results, and Table 3 presents the ANOVA analysis for each pair of groups.

Table 2. Time (in minutes)

Groups	#subjects	Average time	Variance
Control group	8	51.5	257.71
Group #1	6	71.17	322.97
Group #2	6	41.83	36.97

Table 3. Time - ANOVA test

Groups	Total df	F	P-value
Group #1 vs. Control	13	4.65	0.051
Group #2 vs. Control	13	1.93	0.189
Group #1 vs. Group #2	11	14.34	0.003

(Definitions: df = degrees of freedom, F = f test calculated, P-value = the critical value)

As can be seen, the shortest time to complete the task was by group #2, followed by the control group (with no significant difference between them). But the time of group #1 was significantly longest. This is in line with our conjecture. Note the that subjects of group #1 received the mapping rules, which they had to read, understand and follow; this is time consuming, compared to the subjects of the two other groups – in particular because it was the first time that the subjects performed such a task.

Satisfaction
In order to test the satisfaction of the subjects with the method they used, we averaged the two questions of the questionnaire. The results are presented in Tables 4 and 5, and the ANOVA tests in Tables 6 and 7 respectively.

Since in all cases the P-values are bigger than 0.05, we may conclude that there is no significant difference between the levels of satisfaction of the 3 groups. This contradicts our initial conjecture and might indicate that there is need to improve or simplify the instructions and guidelines of the proposed approach.

Table 4. Q1: how easy is it to model with the method you used?

Groups	Count	Average	Variance
Control group	8	3.375	0.84
Experiment group #1	6	3.5	2.7
Experiment group #2	6	3.33	1.87

Table 5. Q2: how do you assess the quality of your solution?

Groups	Count	Average	Variance
Control group	8	3.75	0.21
Experiment group #1	6	3.33	1.87
Experiment group #2	6	3	1.6

Table 6. Q1 - ANOVA test

Groups	Total df	F	P-value
Group #1 vs. Control	13	0.033	0.858
Group #2 vs. Control	13	0.004	0.946
Group #1 vs. Group #2	11	0.036	0.852

Table 7. Q2 - ANOVA test

Groups	Total df	F	P-value
Group #1 vs. Control	13	0.659	0.432
Group #2 vs. Control	13	2.436	0.144
Group #1 vs. Group #2	11	0.192	0.670

5.4 Threat to Validity

As this is an initial attempt to explore the value of the proposed approach, several threats to validity exist. A major limitation of the experiment is the small number of subjects; there is a need to repeat the experiment with more subjects. We also acknowledge that the subjects were not homogenous: graduate students have more experience than third year undergraduates. But the relatively small number of participants did not enable to test statistically the differences in their performance. Another limitation is the limited knowledge and experience of the subjects with respect to graph databases.

A possible limitation is that the experiment was designed, executed, and analyzed by the researchers; this might cause bias in favor of the proposed methods. However, all preparations were done prior to the experiment in order to increase the objectivity.

Lastly, that task of the experiment was relatively small, compared to real world tasks.

Such limitations as detailed above are common in many controlled experiments; but it is well known that real-world controlled experiments are almost impossible to carry out [5].

6 Summary and Future Work

We introduced and evaluated a method for modeling a graph database schema, using a rich ERD that represents the domain of an application. The ERD is mapped in two steps to a graph database schema (GDBS) that preserves the semantics and constraints defined in the original conceptual schema.

In the order to evaluate the proposed method we conducted a controlled experiment in which we compared it with alternative methods, measuring differences in quality of the GDBS created by designers, time to complete the task, and designers' satisfaction. The results of the experiment showed differences in the quality the created GDBS: subjects who used the proposed modeling method produced correct solutions, while subjects who used the ERD only made some errors, but less than the subjects of the control group who used “ad-hoc” method. With respect to the time, we found significant differences: subjects who used the proposed method spent significantly more time than the subjects who used the other two methods.

In the future, we plan to repeat the controlled experiment, using more and homogeneous subjects, more trained with graph databases, having bigger design tasks, perhaps of different size and complexity. Regarding the computation of quality of the designed GDBS, we plan to consider and give proper weights to different errors according to their severity.

We also plan to implement our proposed method, i.e., to create a domain independent tool to be used as plug-in for graph database systems, that will enable to define a (initial) schema of the application. Such implementation might be adopted by graph database system providers. Finally, we intend to extend the proposed schema modeling method to model other types of the NoSQL databases.

References

1. Akoka, J., Comyn-Wattiau, I., Prat, N.: A four V’s design approach of NoSQL graph databases. In: de Cesare, S., Frank, U. (eds.) ER 2017. LNCS, vol. 10651, pp. 58–68. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70625-2_6
2. Angels, R.: A comparison of current graph database models. In: Data Engineering Workshops (ICDEW), pp. 171–177. IEEE (2012)
3. Daniel, G., Sunyé, G., Cabot, J.: UMLtoGraphDB: mapping conceptual schemas to graph databases. In: Comyn-Wattiau, I., Tanaka, K., Song, I.-Y., Yamamoto, S., Saeki, M. (eds.) ER 2016. LNCS, vol. 9974, pp. 430–444. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46397-1_33
4. De Virgilio, R., Maccioni, A., Torlone, R.: Model-driven design of graph databases. In: Yu, E., Dobbie, G., Jarke, M., Purao, S. (eds.) ER 2014. LNCS, vol. 8824, pp. 172–185. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12206-9_14
5. Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., Emam, K.E., Rosenberg, J.: Preliminary guidelines for empirical research in software engineering. IEEE Trans. Softw. Eng. **28**(8), 721–734 (2002)
6. Robinson, I., Webber, J., Eifrem, E.: Graph Databases: New Opportunities for Connected Data. O’Reilly Media, Inc., Sebastopol (2015)
7. Roy-Hubara, N., Rokach, L., Shapira, B., Shoval, P.: Modeling graph database schema. IT Prof. **19**(6), 34–43 (2017)

Author Index