Niels Lohmann, Simon Moser (Eds.)

BPM 2012 Demonstration Track

Demonstration Track of the
10th International Conference on
Business Process Management (BPM 2012)

Tallinn, Estonia, 4 September 2012

Proceedings

Volume Editors

Niels Lohmann
Universität Rostock, Institut für Informatik
18051 Rostock, Germany
niels.lohmann@informatik.uni-rostock.de

Simon Moser
IBM Deutschland Research & Development GmbH
71032 Boeblingen, Germany
smoser@de.ibm.com

Preface

The 10th International Conference on Business Process Management was organized in Tallinn, Estonia, from 3–6 September 2012. The old town of Tallinn and its great scenery provided a wonderful place for the conference. Most of the more than 200 participants also participated in the demo track which took place on 4 September 2012.

In contrast to the previous years, this year's demo track had a slight change in the mode of operation: We decided to start the demonstration track at the end of the first conference day with a teaser session. During this teaser session, the presenters had 90 seconds to advertise their demos. Afterwards, there were five rooms where the individual demos were shown repeatedly for 90 minutes. This mode of operation allowed the conference attendees to switch rooms and watch up to six different tool demonstrations.

Another novelty was the best demonstration award which we proudly awarded during the conference banquet on 6 September 2012 to Anne Rozinat and Christian W. Günther for their excellent work on "Disco: Discover Your Processes".

We would like to thank the authors for their submissions, our Reviewing Committee for their hard work, lively discussions, and for submitting their reviews on time, and the organizers of BPM 2012 conference for their support which made this demo track possible.

All in all, the demonstration track in Tallinn was very successful – the topics covering many fields helped attendees to gain insight into new areas. It continued the tradition to showcase innovative BPM tools originating either from research initiatives or from industry, thus providing an opportunity to present and discuss emerging technologies with researchers and practitioners in the BPM field. Also, included among the presenters were several young scientists, namely, postdocs and students, who brought new perspectives to their fields.

We received 22 demo proposals from 58 authors of 13 countries from which we accepted 10 proposals, making the track again a very competitive event. This proceeding contains the accepted demo proposals. We hope that you will enjoy the work presented in these proceedings and that it will stimulate your thinking and research, and obviously we also hope to meet you all again in next years BPM conference, which will take place in September 2013 in Beijing, China.

November 2012

Niels Lohmann
Simon Moser

Organization

Demo Chairs

Niels Lohmann	Universität Rostock, Germany
Simon Moser	IBM Deutschland Research & Development GmbH

Program Committee

Bob Brodt	Red Hat, Inc.
Christoph Bussler	Xtime, Inc.
Jan Claes	Ghent University
Gero Decker	Signavio
Remco Dijkman	Eindhoven University of Technology
B.F. Van Dongen	Eindhoven University of Technology
Marcelo Fantinato	University of São Paulo - USP
Cédric Favre	IBM Zurich Research Laboratory
Howard Foster	City University London
Luciano García-Bañuelos	University of Tartu
Christian Gierds	Humboldt-Universität zu Berlin
Christian W. Günther	Eindhoven University of Technology
Sandy Kemsley	Kemsley Design Ltd.
Rania Khalaf	IBM TJ Watson Research Center
Oliver Kopp	IAAS, University of Stuttgart
Marcello La Rosa	Queensland University of Technology
Henrik Leopold	Humboldt-Universität zu Berlin
Niels Lohmann	Universität Rostock
Heiko Ludwig	IBM Research
Simon Moser	IBM Deutschland Research & Development GmbH
Hajo A. Reijers	Eindhoven University of Technology
Stefanie Rinderle-Ma	University of Vienna
António Rito Silva	IST/INESC-ID
Nick Russell	Queensland University of Technology
Vishal Saxena	Oracle
Bruno Wassermann	University College London
Barbara Weber	Univ. of Innsbruck
Matthias Weidlich	Technion
Michael Westergaard	Eindhoven University of Technology
Ulrich Winkler	SAP Research Belfast CEC
Moe Wynn	Queensland University of Technology

Additional Reviewers

Reinhold Dunkel
Vishal Saxena

Table of Contents

Demo Papers

BPM Academic Initiative
Fostering Empirical Research

Matthias Kunze, Philipp Berger, and Mathias Weske

Hasso Plattner Institute at the University of Potsdam
Prof.-Dr.-Helmert-Strasse 2-3, 14482 Potsdam,
{matthias.kunze,mathias.weske}@hpi.uni-potsdam.de
philipp.berger@student.hpi.uni-potsdam.de

Abstract. The BPM Academic Initiative strives to support education and research in business process management. This paper announces a platform to be used by researchers to download process models, providing data to be used in empirical research. This paper presents a web portal, where process models can be filtered by various criteria and downloaded, and a research platform that facilitates analysis of the downloaded models by means of a small show case.

1 Introduction

We started the BPM Academic Initiative (BPM AI) together with colleagues from the BPM community in late 2009. The goal of this endeavor has been twofold. First, to support education in business process modeling and analysis by providing a professional software tool free of charge together with assignments to be used by lecturers. Secondly, to strengthen research in our area by showcasing recent research results. Today, the system is used by more than ten thousand students, lecturers, and researchers world wide. At the same time, several research prototypes have been integrated with the platform, including soundness checking, structural analysis of process models, and business process simulation.

In this paper we add to these goals by opening a new service to the BPM community: to provide process models for empirical research in business process management. We hope this service will be as successful as the other parts of the BPM Academic Initiative are today. We are quite confident, since in recent years empirical BPM research has become more and more prominent. However, in many cases researchers find it hard to get access to process models. There are a handful of process model collections that have been used in empirical research on process models. However, these model collections reveal different internal formats, so that researchers need to develop software to access them.

This paper introduces and announces the availability of a platform to filter and to download process models, with accompanying software to process them, such as parsers. In the current version, models of the BPM Academic Initiative collection are provided, but future versions shall also provide additional process model collections. Thereby, researchers are provided with process models and with

software that can help processing them. As a result, researchers can concentrate on their particular research questions, which we hope will strengthen empirical scientific work related to process models and process model collections.

The paper is organized as follows. After a brief discussion of the background of the BPM AI and the offered process models in Section 2, we present a web portal to filter and to download the models in Section 3. Finally, we sketch a show case that is also provided as a short screen cast at `http://vimeo.com/43098307`, in Section 4.

2 Background

The BPM Academic Initiative offers a number of services to support education and research, depicted in Fig. 1: a collection of teaching material, a professional, web-based process model editor and collaboration platform, and the process model provisioning introduced in this paper. We presented the former two components in an earlier demonstration [4], whereas the latter component provides a recent addition to the services of our initiative.

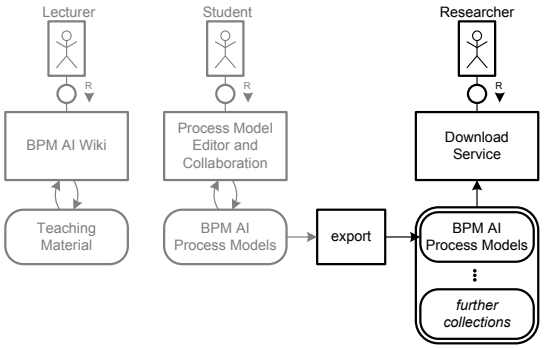

Fig. 1: Overview of services provided by the BPM Academic Initiative

Teaching material made available through the *BPM AI Wiki*[1] comprehensively captures topics of the field of business process management. All materials are offered publicly under the terms of a Creative Commons license and lecturers are invited to contribute to the content. The wiki provides further information about the BPM AI and references related publications.

To facilitate process modeling as part of teaching activities, assignments, or research, a professional, web-based *process modeling platform* from our industry partner Signavio is provided to academics free of charge. The platform sports a holistic set of process modeling languages, e.g., BPMN (including conversation and choreography diagrams), EPC, and Petri Nets. Work from many contributors has made its way into the platform and is offered to the users, as will future

[1] BPM Academic Initiative wiki, cf. `http://bpmai.org`

developments, especially with regard to new scientific features and modeling languages. For instance, LoLa soundness checker [2] and bpstruct [5], a tool to structure spaghetti-like process models automatically, are integrated with the platform; future enhancements are welcome. Additionally, this tool comes with collaboration features that allow multiple process designers to jointly create a process model and discuss revisions by comments.

A multitude of models has been created using the process modeling platform, already. Now, we make these models available for research. Therefore, when signing up, every user of the platform agreed that their models may be reused for empirical research. A subset of these models has already been used towards understanding process modeling [3]; the authors also propose challenging research topics in the context of process model collections. Now, these models are offered to all interested researchers.

3 Service Description

The service to download process models addresses researchers that aim, for instance, at evaluating and validating their research results. As the BPM AI models have been created by students, lecturers, and researchers, they show a high heterogeneity in terms of the used natural language and modeling language, business domain, and quality. Hence, empirical insights that are derived using the BPM AI collection can be assumed to have a high external validity.

By June 2012, over 10000 users have adopted the academic initiative and created over 85000 models that comprise 290000 revisions.

Access and Download Process Models. Process models can be downloaded from a web portal, i.e., no specific software is required to access them.

At `http://bpmai.org/download/`, users are presented with a filter interface, shown in Fig. 2, that allows them to select a subset of models that match following criteria.

Modeling Language allows restricting downloaded models to one or several model types, e.g., BPMN models.

Language denotes the natural language of process models that we derived from the inscriptions of model elements. In particular, research that incorporates labels, e.g., process model similarity, is typically sensitive to the used language.

Connectedness is a quality measure that evaluates the size of the largest connected graph towards the size of the overall model. If a model consists of many disconnected fragments, this measure will be low.

Size counts the number of nodes in a process model and can be used as a simple means to address complexity.

Revisions. As the process model editor creates a new revision of a model each time it is saved, users can choose, whether they want to download all or only the latest revision of the diagrams.

In the filter interface, the user can obtain a random example chosen according to the current filter settings.

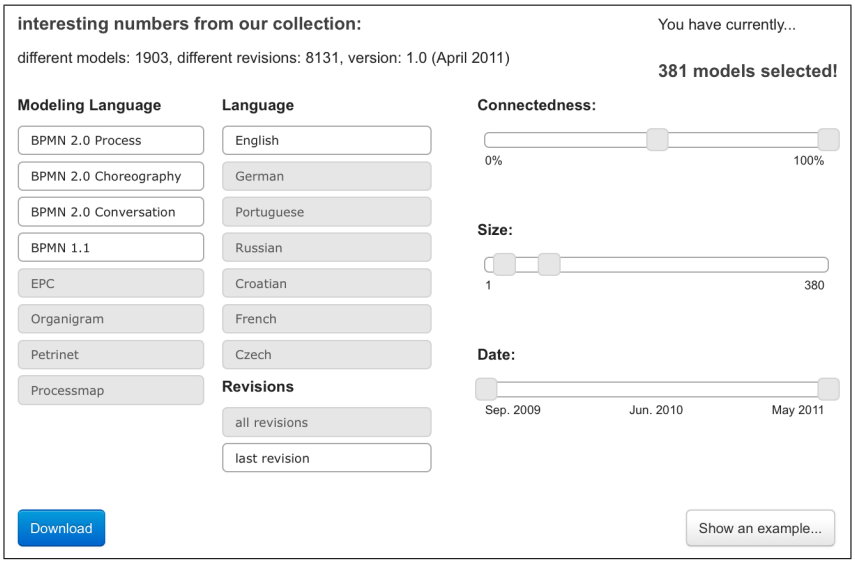

Fig. 2: Interface to filter process models by various criteria.

Once the user clicks the download button, they are presented a form that requests some information, i.e., the researcher's name, email address, affiliation, and a short research proposal to which the models shall contribute. The BPM AI core team will review the research proposal and grant access to earnest requests, thus avoiding abuse of the service. Also, before submitting the request, one has to accept a license agreement that restricts usage of the models to empirical research in non-commercial settings only.

When the request has been granted, the researcher will receive an email with a unique link to download their models. Upon choosing to download, a task will be scheduled in our system that extracts the models selected by the user's filter and creates a zip file. As this may incorporate a significant amount of time, the user will receive an email with the download link.

Research on Process Models and Collections. The downloaded models unzip to a directory structure, and for each process model revision, a JSON file and an SVG file are provided. The JSON file contains the model's structure and attributes, and is used as the internal format of the process modeling tool, whereas the SVG file provides a ready-made vector graphic to display the diagram.

To support researchers in disseminating the directory structure and parse the JSON representation of diagrams, we also offer an open-source platform to process model collections research [1]. This platform provides import functionality for the BPM AI model collection, among others. Once the downloaded models have been imported, the platform offers utilities to filter, transform, and extract information from the process models, similar to the pipes and filter enterprise integration pattern. A mapping for EPC and BPMN models to a generic process model representation is provided, such that features of the jbpt[2] Java library can

[2] jbpt, cf. `http://code.google.com/p/jbpt/`

be used, e.g., workflow graph parsing, net unfolding generation, and to derive behavioral profiles.

4 Show Case

The demo addresses all researchers that focus on process model aspects, in all phases of the business process lifecycle. Models of the BPM AI are generally operative models, i.e., they do not contain technical details as required for enactment or performance evaluation.

In the demonstration, we present a show case that targets at the creation of a word cloud from activity labels in chosen process models. For a short screen cast of this show case, visit `http://vimeo.com/43098307`.

1. In the first part, we explain the capabilities of the user interface to select a subset of process models from the collection. We concisely present the characteristics of the filter criteria.
2. The process of requesting access to the process model collection by providing a research proposal, receiving a response, and downloading the collection is laid out comprehensively. We also discuss the structure of the directories and files included in the downloaded zip file.
3. Finally, we will use the process model collection research platform to extract activity labels from the given process models, independent of their modeling language, and feed them into a word cloud generator. This is intended to show, how researchers can leverage the knowledge of the BPM AI models with small effort.

   A word cloud visualizes the distribution of words in a large set by their size printed on a canvas; an example is depicted in Fig. 3.

Fig. 3: Example word cloud.

References

1. Rami-Habib Eid-Sabbagh, Matthias Kunze, Andreas Meyer, and Mathias Weske. A Platform for Research on Process Model Collections. In *BPMN 2012*, (to appear).
2. D. Fahland, C. Favre, J. Koehler, N. Lohmann, H. Völzer, and K. Wolf. Analysis on demand: Instantaneous soundness checking of industrial business process models. *Data Knowl. Eng.*, 70(5):448–466, 2011.
3. M. Kunze, A. Luebbe, M. Weidlich, and M. Weske. Towards Understanding Process Modeling – The Case of the BPM Academic Initiative. In *(BPMN 2011)*, volume 95 of *LNBIP*, pages 44–58. Springer, 2011.
4. M. Kunze and M. Weske. Signavio-Oryx Academic Initiative. In *BPM 2010 Demo*, volume 615 of *CEUR*, 2010.
5. A. Polyvyanyy, L. García-Bañuelos, and M. Dumas. Structuring acyclic process models. In *BPM'10*, volume 6336 of *LNCS*, pages 276–293. Springer, 2010.

Updatable Process Views for Adapting Large Process Models: The *proView* Demonstrator

Jens Kolb, Klaus Kammerer and Manfred Reichert

Institute of Databases and Information Systems
Ulm University, Germany
{jens.kolb,klaus.kammerer,manfred.reichert}@uni-ulm.de
http://www.uni-ulm.de/dbis

Abstract. The increasing adoption of process-aware information systems (PAISs) has resulted in large process model collections. To support users having different perspectives on these processes and related data, a PAIS should provide personalized views on process models. Especially, changing process models is a frequent use case in PAISs due to evolving business processes or unplanned situations. While process views have been suggested as abstractions for visualizing large process models, no work exists on how to change these models based on respective views. This software demonstration presents the *proView* framework for changing large process models through updates of corresponding process views, while ensuring up-to-dateness and consistency of all other process views related to the changed process model. Respective update operations can be applied to a process view and are correctly propagated to the underlying process model. Furthermore, all views related to this process model are then correctly migrated to its new version as well. Overall, the *proView* framework enables domain experts to evolve large process models over time based on appropriate model abstractions.

Keywords: process model abstraction, process view, process change, view update, process visualization, user-centered process management

1 Introduction

Process-aware information systems (PAISs) provide support for business processes at the operational level [1]. A PAIS strictly separates process logic from application code, relying on explicit *process models*. This enables a separation of concerns, which is a well-established principle in computer science to increase maintainability and to reduce costs of change [2]. The increasing adoption of PAISs has resulted in large process model collections. In turn, each process model may involve different domains, organizational units, and user roles as well as dozens or even hundreds of activities [3]. Usually, the different user roles need customized views on their process models, enabling personalized process

abstraction and visualization [4,5]. For example, managers rather prefer an abstract overview, whereas process participants need a detailed view of the process parts they are involved in [6]. Hence, providing personalized process views is a much needed PAIS feature. A variety of approaches for creating process model abstractions based on process views have been proposed [7,8,9,10]. However, these proposals focus on creating and visualizing views, but do not consider another fundamental aspect of PAISs: change and evolution [11]. More precisely, they do not allow changing a large process model through editing or updating any of its view-based abstractions. As a consequence, process changes still must be directly applied to the core process model, which constitutes a complex as well as error-prone task for domain experts, particularly when confronted with large process models [12]. To overcome this limitation, in addition to view-based process abstractions, users should be allowed to change large process models through updating respective process views. However, this must not be accomplished in an uncontrolled manner to avoid inconsistencies or errors.
The *proView*[1] framework addresses these challenges by providing powerful view-creation operations [13]. The operations allow abstracting process models through the *reduction* and *aggregation* of process elements as well as through changes of the process model notation [14]. In addition, view-update operations allow adapting process views and propagating the respective changes to the underlying process model as well as to other related process views [15]. Our tool presentation will demonstrate these aspects of the *proView* framework in an integrated and comprehensible way.
Section 2 introduces the application scenario we use for our demonstration. Section 3 presents the *proView* framework and the view operations it supports. Section 4 then describes how the application scenario can be supported by using the *proView* framework. Section 5 concludes the paper.

2 Application Scenario

Figure 1 shows a credit request process modeled in terms of BPMN. The process involves human activities referring to three user roles (i.e., *customer*, *clerk* and *manager*) as well as automatic activities executed by the PAIS without user interaction. Assume that the process is started by the customer filling out a credit request form (Step ①). Afterwards, the PAIS checks whether an entry for the customer needs to be created in the CRM system or the customer has been already registered (Step ②). In the latter case, customer information is retrieved from the CRM. Then, the clerk reviews the credit request (Step ③), calculates the risk, and checks the creditworthiness of the customer with the credit protection agency (Step ④). After completing these tasks, he decides whether to reject the request (Step ⑤) or forward it to his manager who finally decides about granting the credit request or not (Step ⑥). If the manager rejects the request, a respective email is sent to the customer (Step ⑦). Otherwise, a confirmation email is sent and the CRM database is updated. Finally, the clerk calls

[1] http://www.dbis.info/proView

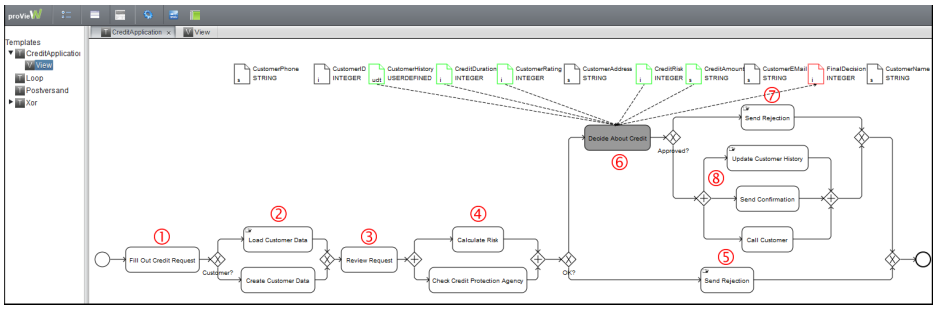

Fig. 1. Credit Application Process

the customer in the context of after sales (Step ⑧), before the process completes. Assume that an evolution of this process model becomes necessary: Before filling out the credit form, the customer shall select the desired credit type. For this purpose, an activity is added by the clerk to the process model. Obviously, this change is relevant for all participants.

The *proView* framework addresses the user-centered visualization and adaptation of large process models. Hence, in the given scenario, it enables personalized views and visualizations of the credit request process for each user role, i.e., the customer, clerk, and manager roles. In particular, the following requirements must be met in order to properly support such a scenario:

R1: It should be possible to provide specific process views on a process model for each user role and to flexibly define those views.
R2: The visual appearance of the process model and process view respectively needs to be flexibly adaptable for each user (role) to meet needs best.
R3: Based on personalized process views and visualizations, elementary model adaptations should be possible, e.g., to insert or delete activities in a user-centered process model (i.e., process view).
R4: In case of changes introduced by a user, all other process views need to be updated to ensure up-to-dateness of all process participants.
R5: Since domain experts hardly have technical process knowledge, high-level operations for creating and adapting user-centered process views are required.

3 proView Framework

Figure 2 gives an overview of the implemented *proView* framework, which consists of two major components: *proViewServer* and *proViewClient*. The *proViewClient* is instantiated for each user and takes care of interactions with the user as well as the visualization of his process models and process views respectively. The proViewClient is based on the *vaadin* web-framework and interacts with the proViewServer using a RESTful communication protocol. The *proViewServer*

implements the logic of the *proView* framework and provides engines for *visualization*, *change*, and *execution & monitoring*. It captures a *business process* through a *Central Process Model (CPM)*. In addition, for a particular CPM, so-called *creation sets (CS)* are defined. Thereby, each CS specifies the schema and appearance of a particular process view [15].
The *visualization engine* generates a process view based on a given CPM and the information captured in a creation set CS, i.e., the CPM schema is transformed to the view schema by applying the corresponding *view-creation operations* specified in CS (Step ⑤). Afterwards, the obtained view schema is *simplified* by applying well-defined *refactoring operations* (Step ⑥). Finally, Step ⑦ customizes the visual appearance of the view (e.g., creating an tree-, form-, or activity-based visualization [8,14]) and delivers it to the proViewClient.

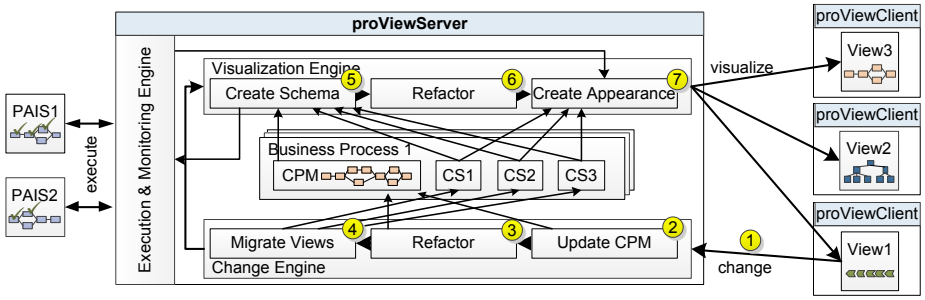

Fig. 2. The proView Framework

When a user updates a view schema, the *change engine* is triggered (Step ①). First, the view-based model change is propagated to the related CPM using well-defined change propagation algorithms (Step ②). Next, the schema of the CPM is simplified (Step ③), i.e., behaviour-preserving refactorings are applied to foster model comprehensibility (e.g., by removing surrounding gateways not needed anymore). Afterwards, the creation sets of all other views associated with the CPM are migrated to the new CPM schema version (Step ④). This becomes necessary since a creation set may be contradicting with the changed CPM schema. Finally, all views are recreated (Steps ⑤-⑦) and presented to users by the proViewClients.

4 proView Demonstration

We revisit our scenario from Section 2 and show how the described requirements can be addressed by *proView*.
Requirement R1: The proViewServer allows creating an arbitrary number of process views by applying aggregation and reduction operations specified in the creation set. Thereby, a *reduction* removes an activity from the respective view, while an *aggregation* combines a set of connected activities to one activity.
Requirement R2: The proViewClient enables users to change the visual appearance of process views, e.g., by switching between the notations provided by

BPMN, ADEPT [16], and proViewForms. The latter allow visualizing process models and views in terms of forms, which support users, not familiar with activity-centered process notations, in understanding complex process logic. Further visual appearances for process views are under construction (e.g., text-based representation).
Requirement R3: The proViewServer provides *view-update* operations which allow inserting and deleting activities as well as AND/XOR branchings [15]. These operations can be applied by an end-user to his process view using the proViewClient and are then be propagated to the proViewServer. Furthermore, *parametrization* of these operations allows for automatically resolving ambiguities when propagating view changes; i.e., change propagation behaviour can be customized. However, at this stage concurrent changes are not enabled in the proViewServer, i.e., only one change at a time is allowed.
Requirement R4: Updates triggered by users are applied to the CPM as well as to associated process views. Their view creation sets are then migrated to the new version of the CPM. Hence, all affected views will be re-created.
Requirement R5: The proViewServer supports high-level operations to create process views. For example, a new view can be created based on the role of a user displaying only those activities he is involved in.
All these aspects are illustrated in our screencast and can be watched at the projects' website: www.dbis.info/proView.

5 Conclusion

In our demonstration, we present the *proView* framework and its operations; *proView* supports the creation of personalized process views as well as the view-based change of business processes, i.e., process abstractions not only serve visualization purpose, but also lift process changes up to a higher semantical level. A set of update operations enables users to update their view and propagate the respective change to the process model representing the overall business process. Finally, we provide migration rules to update all other process views associated with a changed CPM. Similar to this propagation, it can be decided per view, how much information about the change shall be displayed to the user.
The *proView* framework is implemented as a client-server application to simultaneously edit a process model based on views. The implementation of the proView framework has proven the applicability of our approach. Furthermore, user experiments based on the proView demonstrator are planned to systematically analyze whether view-based process changes improve the handling and evolution of large process models. Moreover, the proView demonstrator shall be extended to also execute process views in a PAIS [17]. Overall, we believe that the *proView* framework offers promising perspectives for process participants for evolving their business processes.

References

1. Reichert, M., Weber, B.: Enabling Flexibility in Process-aware Information Systems - Challenges, Methods, Technologies. Springer (2012)
2. Weber, B., Sadiq, S., Reichert, M.: Beyond Rigidity - Dynamic Process Lifecycle Support: A Survey on Dynamic Changes in Process-Aware Information Systems. Computer Science - Research and Development **23**(2) (2009) 47–65
3. Weber, B., Reichert, M., Mendling, J., Reijers, H.A.: Refactoring Large Process Model Repositories. Computers in Industry **62**(5) (2011) 467–486
4. Rinderle, S., Bobrik, R., Reichert, M., Bauer, T.: Businesss Process Visualization - Use Cases, Challenges, Solutions. In: Proc. 8th Int'l Conf. on Enterprise Information Systems (ICEIS'06). Volume 2006., Paphos, Cyprus (2006) 204–211
5. Streit, A., Pham, B., Brown, R.: Visualization Support for Managing Large Business Process Specifications. In: Proc. BPM'05. (2005) 205–219
6. Bobrik, R., Reichert, M., Bauer, T.: Requirements for the Visualization of System-Spanning Business Processes. Proc. DEXA'05 Workshops (2005) 948–954
7. Tran, H.: View-Based and Model-Driven Approach for Process-Driven, Service-Oriented Architectures. TU Wien, Dissertation (2009)
8. Bobrik, R., Bauer, T., Reichert, M.: Proviado - Personalized and Configurable Visualizations of Business Processes. In: Proc. EC-WEB'06. (2006) 61–71
9. Chiu, D.K., Cheung, S., Till, S., Karlapalem, K., Li, Q., Kafeza, E.: Workflow View Driven Cross-Organizational Interoperability in a Web Service Environment. Information Technology and Management **5**(3/4) (July 2004) 221–250
10. Bobrik, R., Reichert, M., Bauer, T.: View-Based Process Visualization. In: Proc. 5th Int'l Conf. on Business Process Management, Brisbane, Australia (2007) 88–95
11. Weber, B., Reichert, M., Rinderle, S.: Change Patterns and Change Support Features - Enhancing Flexibility in Process-Aware Information Systems. Data & Knowledge Engineering **66**(3) (2008) 438–466
12. Reijers, H., Mendling, J.: A Study into the Factors that Influence the Understandability of Business Process Models. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on (99) (2011) 1–14
13. Reichert, M., Kolb, J., Bobrik, R., Bauer, T.: Enabling Personalized Visualization of Large Business Processes through Parameterizable Views. In: Proc. 26th Symposium On Applied Computing (SAC'12), Riva del Garda (Trento), Italy (2012)
14. Kolb, J., Reichert, M.: Using Concurrent Task Trees for Stakeholder-centered Modeling and Visualization of Business Processes. In: Proc. S-BPM ONE 2012, CCIS 284. (2012) 237–251
15. Kolb, J., Kammerer, K., Reichert, M.: Updatable Process Views for User-centered Adaption of Large Process Models. In: Proc. Intl. Conf. on Service Oriented Computing (ICSOC'12), Shanghai, China (2012) to appear
16. Dadam, P., Reichert, M.: The ADEPT Project: A Decade of Research and Development for Robust and Flexible Process Support. Computer Science - Research and Development **23**(2) (April 2009) 81–97
17. Kolb, J., Hübner, P., Reichert, M.: Automatically Generating and Updating User Interface Components in Process-Aware Information Systems. In: Proc. 10th Int'l Conf. on Cooperative Information Systems (CoopIS'12). (2012) to appear

The Shared Process Model

Cédric Favre, Jochen Küster, and Hagen Völzer

IBM Research - Zurich, Switzerland,
{ced,jku,hvo}@zurich.ibm.com

Abstract. Maintaining different but consistent views on the same process is often necessary in BPM projects. For example, a business analyst typically works on a business process model at a different level of abstraction than an IT architect. These views evolve independently and synchronizing them is not trivial. In this demonstration, we showcase our Shared Process Model prototype that allows different stakeholders to work on different views of a business process model while keeping these views synchronized. In particular, we will look at scenarios where a business view and an IT view are modified and a subset of the modifications need to be propagated from one view to the other. This demonstration targets the general BPM audience interested in ensuring consistency between various level of realisation of a business process model and solving the related round tripping problems. This demonstration will also appeal to people interested in process comparison and process merging — the two core techniques used by our prototype to propagate changes from one view to the other.

1 Relevance to BPM field

A business process model is used by different stakeholders for different purposes. For example, a business analyst uses a business process model to document, analyze and communicate a process while an IT architect uses a process model to implement the process on a particular process engine. Both stakeholders use a model that represents the same process but from a different perspective which has different requirements. For example, the IT architect is interested in modeling the service invoked when a task is executed and the exception flow triggered when the service invocation fails. For the business analyst, these implementation details clutter the process and therefore should not appear in his view. Note that the differences are not only IT refinements, i.e., the business view is not just a subset of the IT view. We study the differences between the two perspectives in more detail elsewhere [4].

In this demo, we will showcase our *Shared Process Model* prototype. The Shared Process Model supports parallel maintenance of different views of a the same BPMN 2.0 model, a capability lacking in major BPM suites [2]. In Sect. 2, we discuss the features, the supported scenarios, and an overview of the implemented approach of our prototype. In Sect. 3, we describe a scenario that we will use as screen-cast to highlight the features of the Shared Process Model. In Sect. 4, we conclude with the limitations of the prototype and future work.

2 The Shared Process Model

The Shared Process Model is a research prototype built on top of the *BPMN2 Modeler* — an Eclipse-based graphical BPMN 2.0 model editor [1]. The Shared Process Model provides to the users two different views on a process, a *business view* and an *IT view* as illustrated by Fig. 1.

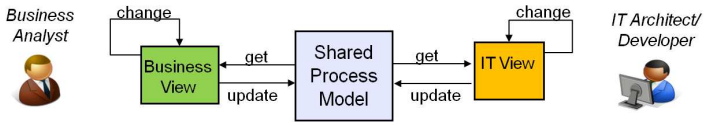

Fig. 1: The Shared Process Model

Supported operations: The Shared Process Model supports the independent development of a business-view and an IT view of a process through tree key operations:

get returns a copy of the current version of the IT or business view,
change allows the user to modify his copy of the IT or business view, and
update allows the user to synchronize his copy of the IT or business view with the Shared Process Model. Each change can be either designated as a *common* or a *private* change. A common change is automatically propagated by the Shared Process Model to the other view whereas a private change is not.

Central features: The Shared Process Model allows a user to modify either the business or the IT view and to propagate a subset of these modifications to the other view. For example, the IT architect might decide to update changes made to the main control-flow of the process as common but to keep private the addition of the exception flow. The IT architect may need to propagate changes to the business view because the initial business model is incomplete, contains modeling errors, contradicts some IT requirements, or does not faithfully represent the actual business process. Propagating changes is also required when the business or the IT requirements change. The reasons and frequency of these updates are presented in more detail in a technical report [4].

It also ensures that the two views remain *consistent*, i.e., the IT view is a 'faithful' representation of the business view and vice et versa. The enxact notions of consistency considered ad how they are ensured or checked is out of scope of this paper but are presented in the technical report [4].

Finally, the Shared Process Model provides a set of model *refactoring operations* to support the user in modifying a view while retaining its consistency with the other view. For example, it provides refactoring operations to refine an activity into a subprocess or into a set of activities together with the control-flow between them, to specify that an activity is implemented as a script task or a service task, and to simplify a portion of the process into a single activity.

Shared Process Model implementation and change propagation: Fig. 2 illustrates the internal of our implementation a Shared Process Model: two BPMN 2.0 process models representing the two views and *correspondences* (the highlighted vertical arrows) relating the nodes of the two models. Note that correspondences can point to multiple corresponding nodes and that some nodes of the IT view do not have a corresponding node in the business view.

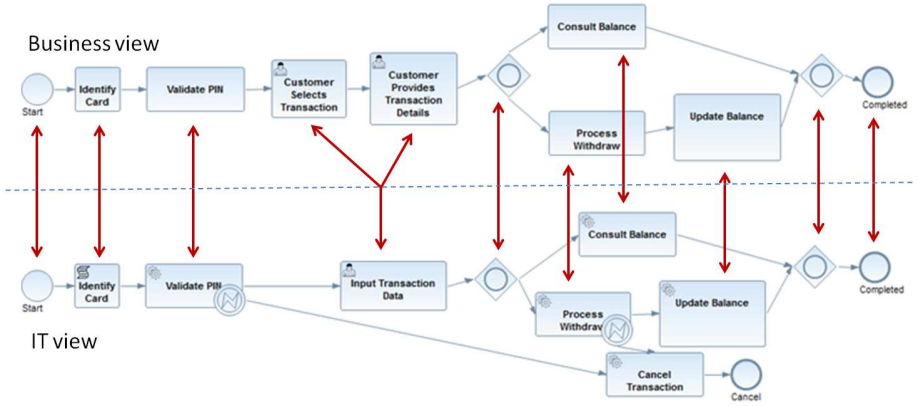

Fig. 2: A example of Shared Process Model internals

When a user modifies one view, let us say the IT view, and updates the Shared Process Model, we compute the changes between the updated IT view and the Shared Process Model version of the IT view (which represent the view before the modifications) using a comparison framework [5]. The common changes, formulated for the IT view, are translated into changes applicable to the business view. The translation is based on the correspondences and, when navigating correspondences with multiple targets in the business view, it uses structural analysis on the business view. Finally, the common changes are merged using a merging framework that we developed.

3 The Shared Process Model in Action

We now present a short scenario where the Shared Process Model is used to synchronize a business and an IT view on a process. A screen-cast of featuring this scenario is available on the project webpage [3]. This scenario features two actors Alan, a business analyst and Paul, an IT architect.

Initialization of the Shared Model First, Alan captures the business process model illustrated by Fig. 3. Alan initializes the Shared Process Model, which now contains this process in both views and the appropriate correspondences. From now on, Alan will work on the business view and Paul on the IT view. Alan asks Paul to create an implementation of the process model.

Fig. 3: Initial business process model

Private and Common Changes: Using refactoring operations, Paul refines the specification of the activities by specifying their realization, some activities are implemented as script tasks other by service tasks. He also adds the exception flow. These changes are only relevant to the IT view. Therefore, Paul commits the changes as private. Paul then realizes that he can optimize the control-flow of the process and that Alan forgot one activity. These changes are relevant to business view and Paul updates them as common. The IT view now displays the process in Fig. 4 while the business view displays the process illustrated on top of Fig. 2.

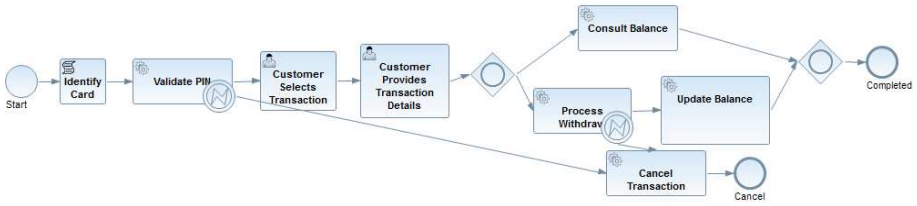

Fig. 4: IT view after private and common changes

Private refactoring: Paul wants to convert the two human activities into one. He uses the *simplify selection* refactoring wizard, which turns a selection of elements into a single activity and reroutes the incident edges of the selection accordingly. The refactoring also creates the correspondence between the two business activities and the IT activity. This change is a private change. The Shared Process Model now contains the two views illustrated by Fig. 2.

Common changes using correspondences Finally, Paul inserts a new activity between 'Validate Pin' and 'Input Transaction Data' as a common change which results in the IT view illustrated by Fig. 5. Looking at Fig. 2, one realizes that the translation of this change, as described in Sect. 2, requires to navigate the correspondences and to perform a structural analysis.

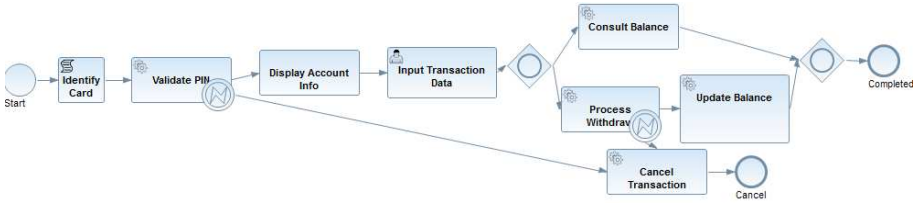

Fig. 5: IT view after selection simplification and insertion of task

Updating this change results in the business view illustrated by Fig. 6. The two views have evolved and are now significantly different. However, the Shared

Process Model is still able to propagate automatically changes between the two views ensuring that they stay consistent maintaining the correspondences between its different elements.

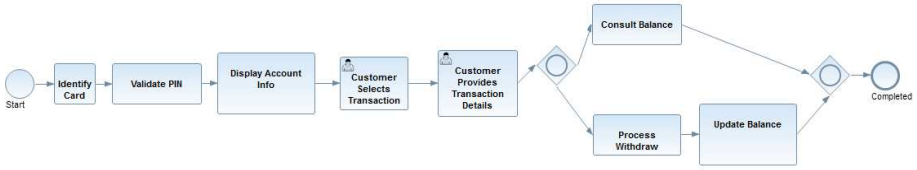

Fig. 6: Final business view

4 Limitations and future work

In this prototype, we focused on the propagation of control-flow related modifications. The propagation of some attribute modifications such as, for example, changing the type of an event is not implemented but could easily be added. We also consider the implementation of a user interface allowing a user to approve or reject the changes made by another user. For example, Alan could approve only a subset of the common changes proposed by Paul.

We only presented the modifications of two views: business and IT. The Shared Process Model can be generalized to any two BPMN 2.0 views. The architecture of the prototype as well as comparison and merging components would scale up to a larger number of views. However, the correspondences would require a more complex representation and management.

We currently support scenarios where IT view and business view modifications are interleaved. Ultimately, we aim to integrate the Shared Process Model in a modern BPM suite where models sit in a shared repository, the Shared Process Model would then be a shared object in the repository and new scenarios would involve concurrent editing of the IT and business view. Support to prevent, detect, and resolve conflicts arising from concurrent editing is necessary for these scenarios.

References

1. The BPMN 2.0 Modeler available at `http://eclipse.org/projects/project.php?id=soa.bpmn2-modeler`
2. M. Castelo Branco, K. Czarnecki, J. Kuester, and H. Völzer. An Empirical Study on Consistency Management of Business and IT Process Models. available at `http://gsd.uwarterloo.ca/reportstudybpm`
3. C. Favre, J. Kuester, and H. Völzer. The Artifact Consistency Management project: `http://researcher.watson.ibm.com/researcher/view_project.php?id=3210`
4. J. Kuester, H. Völzer, C. Favre, M. Castelo Branco, and K. Czarnecki. Supporting Different Process Views through a Shared Process Model. Technical report, IBM Research, RZ3823.
5. J. Küster, C. Gerth, A. Förster, and G. Engels. Detecting and resolving process model differences in the absence of a change log. *Business Process Management*, pages 244–260, 2008.

Eventifier: Extracting Process Execution Logs from Operational Databases

Carlos Rodríguez[1], Robert Engel[2], Galena Kostoska[1], Florian Daniel[1], Fabio Casati[1], and Marco Aimar[3]

[1] University of Trento,
Via Sommarive 5, I-38123, Povo (TN), Italy
{crodriguez,kostoska,daniel,casati}@disi.unitn.it
[2] Vienna University of Technology
Institute of Software Technology and Interactive Systems
engel@ec.tuwien.ac.at
[3] Opera21 Group SpA,
Rovereto (TN), Italy
maimar@opera21.it

Abstract. This demo introduces *Eventifier*, a tool that helps in reconstructing an event log from operational databases upon which process instances have been executed. The purpose of reconstructing such event log is that of discovering process models out of it, and, hence, the tool targets researches and practitioners interested in process mining. The aim of this demo is to convey to the participants both the conceptual and practical implications of identifying and extracting process execution events from such databases for reconstructing ready-to-use event logs for process discovery.

1 Introduction

Process discovery is the task of deriving a process model from process execution data that are typically stored in event logs, which in turn are generated by information systems that support the process execution [5]. Most of the approaches available in the state of the art assume the existence of an event log, where each event is assumed to have information, such as a process name, activity name, execution timestamp, event type (e.g., start or end), and process instance ID. In practice, most companies do not really have such an event log, either because they do not have a business process engine that is able to generate such logs or, if they do, the engine supports only parts of the process, e.g., because parts of the process are supported by legacy systems. In the second case, it may also happen that the engine does not generate an event log that can be used for process discovery, e.g., if the log contains only events regarding errors in the system.

The information stored in an event log commonly provides a very narrow and focused view on the overall data produced by a process during its execution (e.g., focusing on errors for recovery or control flow decisions and actors for

auditing). Typically, however, an information system also stores the full data produced by a process inside its ***operational databases*** (OD) (also known as *production databases*), where these data comprise process progression data, process state data, business data produced throughout the process, data related to the regular operations of an organization, as well as their related business facts and objects [2]. ODs therefore store more and richer data than event logs, but blur different aspects of data and neglect the event-based nature of process executions. For this reason, process discovery starts from event logs.

With this demo, we approach the problem of producing process execution events in a fundamentally different context, i.e., in a context where we do *not* have access to the information system running the process (hence we cannot instrument it) and where the only way of obtaining process execution events is deriving them from the OD of the information system *after* the actual process execution. We call this activity ***eventification*** of the OD and we perform it with the help of our tool ***Eventifier***. For the rest of the paper, we assume that the OD is a ***relational database*** [4].

Significance to the BPM field. Much attention has been paid so far to the problems of representing event logs [6], event correlation [3] and process discovery [5], while the problem of how to produce good events has been neglected by research. As explained above, ***Eventifier*** approaches an important issue in the field of process mining by providing an application that will help both researches and practitioners working in the field.

2 Eventification of the Operational Database

Let's start by giving some preliminary definitions. An ***event log*** can be seen as a sequence of events $E = [e_1, e_2, ..., e_m]$, where $e_i = \langle id, tname, pname, piid, ts, pl \rangle$ is an event of a process instance, with *id* being the identifier of the event, *tname* being the name of the task the event is associated with, *pname* being the name of the process type, *piid* being the process instance identifier, *ts* being the timestamp of the event, and *pl* being the payload of the event. Thus, an event log stores traces of process executions as atomic events that represent process progression information and that may carry business data in their payload.

Reconstructing an event log E with events e_i means deciding when to infer the existence of an event from the data in the OD and filling each of the attributes of the event structure with meaningful values. These values either stem from the data in the OD or they may be provided by a domain expert. Specifically, for the *id* attribute, assigning an identifier to an event means recognizing the *existence* of the event. Given that we do not have real events in the OD but other, indirect evidence of their occurrence, there is no "correct" or "original" event identifier to be discovered. The question here is what we consider *evidence* of an event. Similarly, in the case of *tname*, without the concept of task in the applications of the information system, there is no explicit *task naming* that can be discovered from the data. Thus, we need to find a way to label the boxes that

will represent tasks in the discovered model. The value for the attribute *pname* (the *process name*) we can only get from the domain expert, who knows which process she is trying to discover. Then, the process instance identifier (*piid*) is needed to group events into process instances. The *piid* is derived by means of event correlation based on the values of the attributes of the identified events. The attribute *ts* is needed to *order* events chronologically, which is a requirement for process discovery. Therefore, we need to find evidences in the OD that help us in determining the ordering of events. Finally, the goal of choosing a payload *pl* for the purpose of eventification is not to reconstruct the complete *business data* that can be associated with a given task or event, but rather that of supporting the correlation of events into process instances. We can get this data from the rows that originate the events.

We call the assignment of values to *id*, *pname* and *tname* the *identification* of an event, to *ts* the *ordering* of events, to *pl* *data association*, and to *piid* *correlation*. These four activities together constitute the ***eventification*** process, and it is helped by heuristics in the form of ***eventification patterns***:

Event identification patterns. These patterns help in the identification of events from the OD. In these patterns, we assume that the existence of a row in a relation R indicates the presence of an event. We express these patterns as a function:

$$identify(R, pname, tname) \rightarrow e^0 = \langle id, pname, -, tname, -, t \rangle$$

where *pname* and *tname* are defined by the domain expert, and t is the tuple in R that originated e^0. In concrete, we rely on the following three patterns for the identification of events:

- *Single row, single event pattern* (Figure 1(a)). In this pattern, each row in a relation R indicates the existence of an event. R can be obtained with a simple SQL query as:
  SELECT * FROM $r_1, r_2, ..., r_n$
  WHERE [JOIN conditions for $r_1, r_2, ..., r_n$];
- *Single row, multiple event pattern* (Figure 1(b)). A tuple in R can evidence the existence of more than one event, such as when different values of the attributes A_i of R indicate different potential events. In this case, the relation R is built by applying filtering conditions in the WHERE clause so as to keep only the target events:
  SELECT * FROM $r_1, r_2, ..., r_n$
  WHERE [JOIN conditions for $r_1, r_2, ..., r_n$]
  AND [filtering conditions for the target event, e.g., $r_2.dispatched = yes$];
- *Multiple row, single event pattern* (Figure 1(c)). Multiple rows in a relation R indicate the presence of a single event. This last pattern is useful, for instance, when we deal with a *denormalized* relation that mixes data at different granularities, e.g., when in a single tuple we find both the header of an invoice and the item sold. The SQL for R has the following form,
  SELECT DISTINCT $A_1, A_2, ..., A_k$ FROM $r_1, r_2, ..., r_n$
  WHERE [JOIN conditions for $r_1, r_2, ..., r_n$] ;

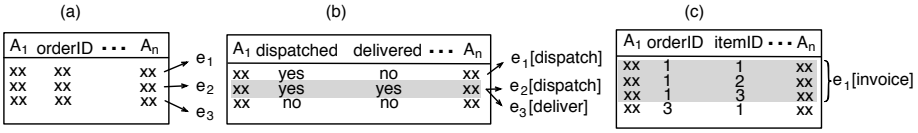

Fig. 1. Types of event identification patterns: (a) single row, single event, (b) single row, multiple events, and (c) multiple row, single event pattern

where the attributes A_i should be the higher granularity attributes that would be typically used in a GROUP BY, SQL statement.

Event ordering pattern. The event ordering pattern aims at deriving the ordering of events from time-related information associated to the records stored in the OD, and is represented as:

$$order(e^0) \rightarrow e^1 = \langle id, pname, -, tname, ts, t \rangle$$

where e^1 is the result of attaching a timestamp value to ts, and ts is the projection of all timestamp or date attributes of $e^0.t$ generated by the previous pattern. If only one timestamp can be found, it is used straightaway. If there are more possible timestamps in pl, the domain expert chooses the one that best represents the execution time of the task.

Data association pattern. The data association pattern aims to select which data to assign to pl. In the above patterns, we have so far simply carried over the complete row t as payload of the event, while here we aim to select which attributes out of the ones in t are really relevant. Our assumption is that all necessary data is already present inside t, that is, we do not need to consult any additional tables of the OD to fill pl with meaningful data. Thus, in the event identification step, the necessary tables are joined, and t contains all potentially relevant data items. The data association pattern is represented as:

$$getdata(e^1) \rightarrow e^2 = \langle id, pname, -, tname, ts, pl \rangle$$

where e^1 is as defined before, and pl is the new payload computed by projecting attributes from t. In absence of any knowledge about the OD by the domain expert, the heuristic we apply is to copy into pl all attributes of t, except timestamps and auto-increment attributes, which by design cannot be used for correlation. The domain expert can of course also choose manually which attributes to include and which to exclude.

Event correlation patterns. Eventually, we are ready to correlate events and to compute the *piid* of the identified events. The goal of event correlation is to group events into process instances, which are the basis for process discovery. As explained above, we assume that after associating the final payloads to events all information we need to correlate events is present in the payload pl of the events in the form of attribute-value pairs. In practice, correlating events into traces means discovering the mathematical function over the attributes of pl that tells

if an event belongs to a given process instance, identified by the output *piid* of the function. We represent this step as follows:

$$correlate(e^2) \rightarrow e = \langle id, pname, piid, tname, ts, pl \rangle$$

where e^2 is as defined above and e is the final version of the discovered event from the OD with the attribute *piid* filled with a suitable identifier of the process instance the event belongs to.

3 The Eventifier Environment

Figure 2 provides an architectural view on the resulting approach to eventification, which is a semi-automated process that requires the collaboration of a domain expert having some basic knowledge of the OD to be eventified. First, the domain expert identifies events in the OD, orders them, and associates data with them. All these activities are supported the the so-called *Event Extractor*, which supports the domain expert in an interactive and iterative fashion. The result of this first step is a set of events, which are however not yet correlated. Correlation is assisted via a dedicated *Event Correlator*, which again helps the domain expert to interactively identify the best attributes and conditions to reconstruct process traces. The result of the whole process is an event log that is ready for process discovery.

The Eventifier is implemented as an integrated platform that includes the components for eventification, correlation and process discovery. These components allow domain experts to interactively apply patterns and to navigate end-to-end from the OD to the discovered process model and back. Since our aim is not to make contributions on process discovery, we use existing process discovery algorithms implemented as plugins for the popular process mining suite ProM [6]. All components are implemented as Java desktop applications using standard libraries such as Swing. The implementation of the Event Correlator is partly based upon a software tool originally developed for the correlation of EDI messages [1]. For the creation of XES-conformant event logs [6] that are used in the interface to process discovery in ProM, we employ the OpenXES libraries (`http://www.xes-standard.org/openxes/start`). Figure 3 shows the screenshots of the Event Extractor and Correlator components.

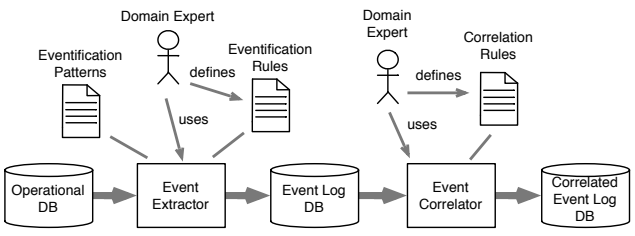

Fig. 2. Overview of the database eventification prototype and approach.

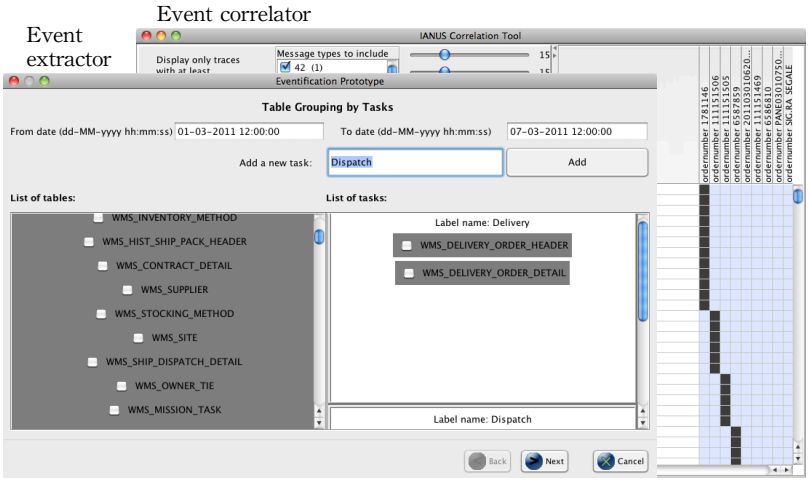

Fig. 3. Screenshots of the components of our integrated platform for eventification.

4 Demo scenario

A demo video of our eventification tool in action can be found at the website `http://sites.google.com/site/dbeventification`. The demo is in the form of a screencast and illustrates the main features of our tool using as scenario the case of an Italian logistics company for refrigerated goods. In this video we clearly show the two main tasks of our approach as outlined in Figure 2 and we also show the final outcome in terms of the process model discovered from the reconstructed event log.

Acknowledgements. This work was supported by the Ianus project funded by the Province of Trento (Italy) and Opera21 Group and by the Vienna Science and Technology Fund (WWTF) through project ICT10-010.

References

1. R. Engel, W. van der Aalst, M. Zapletal, C. Pichler, and H. Werthner. Mining Inter-organizational Business Process Models from EDI Messages: A Case Study from the Automotive Sector. In *24th Int. Conf. on Advanced Information Systems Engineering (CAiSE 2012)*, LNCS 7328, pp.222-237. Springer, 2012.
2. R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. Wiley, 2002.
3. H. Montahari-Nezhad, R. Saint-Paul, F. Casati, and B. Benatallah. Event Correlation for Process Discovery from Web Service Interaction Logs. *VLDB Journal*, 20(3):417–444, 2011.
4. R. Ramakrishnan and J. Gehrke. *Database Management Systems*. McGraw-Hill, 2007.
5. W. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
6. H. Verbeek, J. Buijs, B. van Dongen, and W. van der Aalst. XES, XESame, and ProM 6. In *Information Systems Evolution*, volume 72, pages 60–75. 2011.

Supporting Blended Workflows

Davide Passinhas[1], Michael Adams[2], Bernardo Oliveira Pinto[1], Ricardo Costa[1], António Rito Silva[1], and Arthur H.M. ter Hofstede[2,3]

[1] INESC-ID/IST/Technical University of Lisbon, Av. Prof. Dr. Cavaco Silva, 2744-016 Porto Salvo, Portugal
{davide.passinhas,bernardo.pinto,ricardo.antunes.costa,rito.silva}@ist.utl.pt

[2] Queensland University of Technology, Brisbane, Australia
{mj.adams,a.terhofstede}@qut.edu.au

[3] Eindhoven University of Technology, Eindhoven, The Netherlands

Abstract. Despite advances in the field of workflow flexibility, there is still insufficient support for dealing with unforeseen exceptions. In particular, it is challenging to find a solution which preserves the intent of the process as much as possible when such exceptions are encountered. This challenge can be alleviated by making the connection between a process and its objectives more explicit. This paper presents a demo illustrating the blended workflow approach where two specifications are fused together, a "classic" process model and a goal model. End users are guided by the process model but may deviate from this model whenever unexpected situations are encountered. The two models involved provide views on the process and the demo shows how one can switch between these views and how they are kept consistent by the blended workflow engine. A simple example involving the making of a doctor's appointment illustrates the potential advantages of the proposed approach to both researchers and developers.

1 The Blended Workflow Approach

The blended workflow engine supports a novel approach [1] to workflow management systems that provides end users with two views of the same workflow instance, a "classical" workflow view based on an explicit process model and a view which shows objectives and their relationships. By executing the workflow instance according to the former view, end users' work is guided by a definition of what is the standard behaviour, specified by explicit organisational rules, whereas the latter view empowers end users to use their tacit domain knowledge to handle unexpected situations.

The two views are supported by two different specifications, one activity-based and the other goal-based. Both specifications share a common data model. As a workflow instance progresses toward its completion via consecutive instantiations of its data model, the two specifications describe a prescriptive and a descriptive way of instantiating the data model.

The two specifications are consistent from a semantic point of view: any workflow instance that is executing according to one of the specifications can continue executing according to the other. Therefore each view can present the current state of the workflow instance to end users, who can switch between the two views without having to redo any of the work they have done while using the other view. However, it is possible to produce more information using the goal-based specification, so that additional knowledge that may be useful when dealing with unexpected situations can be acquired. Additionally, it is possible to relax some restrictions when executing according to the goal view or to skip the execution of activities in the activity view.

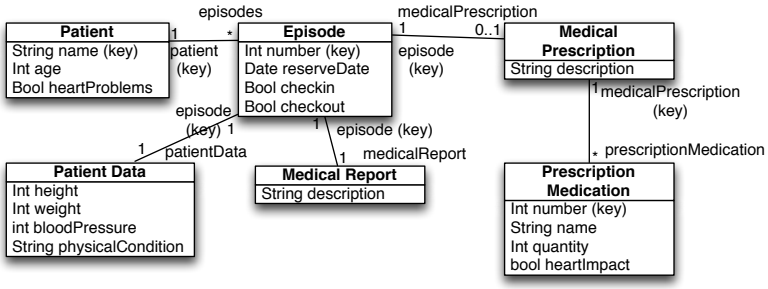

Fig. 1. Doctor Appointment Example - Data Model

Figure 1 presents the data model for a Doctor Appointment workflow specification. The creation of a workflow instance corresponds to the creation of an `Episode` instance and its association to an existing `Patient` instance. The workflow instance progresses by creating instances of the other entities. Eventually a `Medical Report` will be written and the patient checks out.

Figure 2 presents the activity-based specification of the Doctor Appointment example. The specification follows a typical BPMN[4]-like activity-based specification enriched with pre- and post-conditions, denoted respectively by PRE and POS. For each activity, its pre- and post-conditions describe what should be the data model state immediately before and after the execution of the activity, respectively. An activity-based specification can execute in an activity engine in isolation, ignoring its set of pre- and post-conditions, but pre- and post-conditions are necessary for the blended workflow to keep both views synchronized, as we shall show in the demo.

The specification shows that the `Booking` activity creates an instance of `Episode` and sets its `reserveDate` attribute. Note that the creation of an `Episode` instance requires a `Patient` instance and a value for the `number` attribute, which

[4] http://www.bpmn.org/

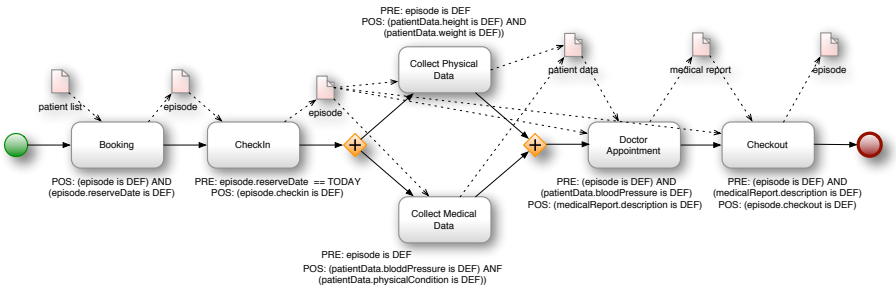

Fig. 2. Doctor Appointment Example - Activity Specification

constitute the key attributes of `Episode`. As another example, note that the execution of `Collect Medical Data` only requires an instance of `Episode`. This relaxed restriction occurs because the blended workflow allows activities to be skipped. In this case it would be possible to skip the `Booking` and `Checkin` activities, but to execute `Collect Medical Data` if an `Episode` instance is created.

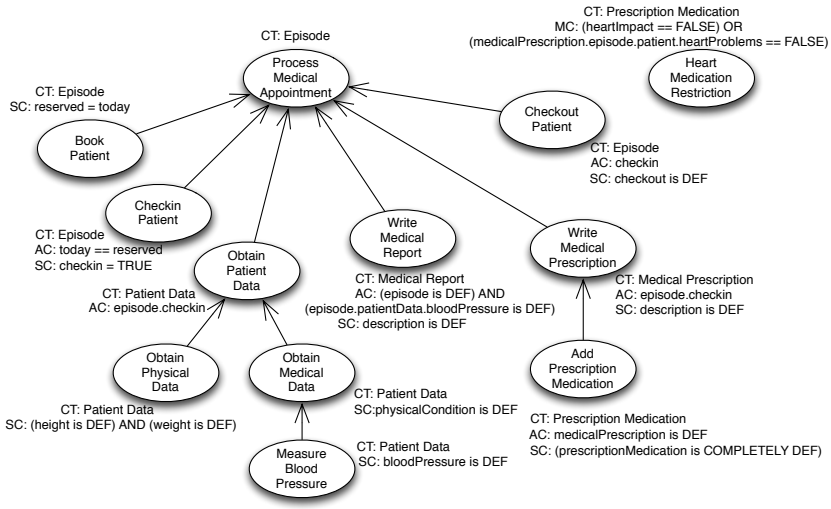

Fig. 3. Doctor Appointment Example - Goals Specification

Figure 3 presents the goal-based specification that adapts from [2] two kinds of goals: achievement and maintain. Achievement goals are specified in a tree and have activation conditions (AC), which should hold true to allow the goal to

become active, and success conditions (SC), which define when a goal is achieved. To achieve a goal it is also necessary that all its sub-goals have succeeded. Additionally, a goal has a context (CT) that represents the data model entity which is the root of its condition's evaluation. A maintain goal describes a data model invariant.

The achievement goal `Write Medical Report` has the `Medical Report` entity as context, and requires both an `Episode` instance and a value for the patient's blood pressure to become active. Upon the goal achievement, the `description` attribute is defined. The maintain goal `Heart Medication Restriction` states that it is not possible to prescribe drugs that may adversely impact the hearts of patients with heart problems.

When executing a workflow instance in the activity view end users can: (1) execute an activity; or (2) skip the execution of an activity. In both situations the process instance progresses according to the control-flow definition. If an activity's pre-condition does not hold when it is enabled according to the control-flow specification, the end user can execute a pre-activity to satisfy the pre-condition.

When executing a workflow instance in the goal view end users can: (1) activate a goal; (2) achieve a goal; (3) skip a goal; (4) redo a goal; (5) disable a goal's activation condition; (6) disable a maintain goal; or (7) create a new goal. When a goal becomes active, its sub-goals also become active. If a goal's success condition holds when it is activated, then it is immediately achieved. When a goal is achieved, the activity view evaluates whether any of the enabled activities can complete. The redo of a goal preserves the state: the same set of conditions continue to hold.

2 Demo Script

In this section we describe the demo script and its objectives. For a screen-cast please visit URL: `http://www.youtube.com/watch?v=Anb4kuXtBgc`. The tool does not include a worklist manager that assigns workitems to users based on their roles because it is not relevant for the blended workflow proof of concept. Therefore in the demo the doctor is able to execute any workitem, either goal or activity.

Second Opinion case

1. Start a Doctor Appointment process for Patient Davide Passinhas;
2. In the activity view, execute all activities until `Doctor Appointment` activity is enabled;
3. An unexpected situation occurs: during the `Doctor Appointment` activity the patient's condition deteriorates, and the doctor decides to measure his blood pressure again: (1) the doctor switches to the goal view, activates goal `Measure Blood Pressure` and redoes it;

4. Due to this situation the doctor decides to seek a second opinion from a colleague, so in the goal view she: (1) creates a new entity, `Second Opinion`, defines a relationship with entity `Medical Report`, and then creates a new goal, `Second Opinion`, which has a success condition requiring a `Second Opinion`'s `description` attribute to be defined; (2) activates goal `Write Medical Report`, which also activates goal `Second Opinion`; (3) goal `Second Opinion` is enabled by the blended workflow engine; (4) goal `Second Opinion` is achieved by the user; (5) goal `Write Medical Report` is enabled by the blended workflow engine; and (6) goal `Write Medical Report` is achieved by the user;
5. The doctor switches to the activity view and verifies that the `Doctor Appointment` activity is completed, and the `Checkout` activity is enabled for execution;
6. The doctor decides to prescribe a medication. She switches again to the goal view: (1) activates goals `Write Medical Prescription` and `Add Prescription Medication`, with the latter being enabled by the blended workflow engine; (2) achieves goal `Add Prescription Medication`; (3) activates new goal `Add Prescription Medication`, and prescribes a medication that may adversely impact on the heart; (4) tries to achieve goal `Add Prescription Medication` with the prescribed medication, but receives an error message because the maintain goal `Heart Medication Restriction` is violated; (5) the doctor decides to take the responsibility to prescribe the drug anyway, and so disables the maintain goal `Heart Medication Restriction`; and (6) executes goal `Add Prescription Medication`;
7. The doctor switches to the activity view and executes the `Checkout` activity, finishing the process.

Urgency case

1. Start a Doctor Appointment process for Patient David Martinho;
2. The `Booking` activity is executed and the `reservedDate` is set for 10 days later;
3. Due to urgency of the patient's condition, the patient requires an examination by the doctor before the `reserveDate` but the `Checkin` activity is not enabled for execution;
4. The doctor switches to the goal view, activates the `Checkin Patient` goal and overrides the goal's activate condition, '`today == reserveDate`', so that the goal can become enabled;
5. The doctor executes the `Checkin Patient` goal. The blended workflow engine automatically completes the `Checkin` activity and enables activities `Collect Physical Data` and `Collect Medical Data`;
6. In the activity view `Collect Physical Data` and `Collect Medical Data` activities are skipped by the doctor, due to the urgency;
7. The doctor executes the pre-activity in the `Doctor Appointment` activity to fill in the missing data, the blood pressure data, and continues execution.

The current version of the tool is a second prototype that extends a first prototype [3] implemented one year ago. It was implemented in JAVA using the FénixFramework[5] for the domain specification and persistency, Vaadin[6] for the user interfaces, and the YAWL [4] engine to execute the activity specification. The code is publicly available in a GitHub repository[7].

3 Acknowledgement

The research was partially supported by the Portuguese Foundation for Science and Technology, FCT-MCTES (INESC-ID multiannual funding) through the PIDDAC Program funds.

References

1. Silva, A.R.: A blended workflow approach. In Daniel, F., Barkaoui, K., Dustdar, S., Aalst, W., Mylopoulos, J., Rosemann, M., Shaw, M.J., Szyperski, C., eds.: Business Process Management Workshops. Volume 99 of Lecture Notes in Business Information Processing., Springer (2012) 25–36
2. Van Lamsweerde, A.: Goal-oriented requirements engineering: A guided tour. In: Proceedings of the Fifth IEEE International Symposium on Requirements Engineering. RE '01, Washington, DC, USA, IEEE Computer Society (2001) 249–262
3. Pinto, B.O., Silva, A.R.: An architecture for a blended workflow engine. In Daniel, F., Barkaoui, K., Dustdar, S., van der Aalst, W., Mylopoulos, J., Rosemann, M., Shaw, M.J., Szyperski, C., eds.: Business Process Management Workshops. Volume 100 of Lecture Notes in Business Information Processing., Springer (2012) 382–393
4. ter Hofstede, A.H.M., van der Aalst, W.M.P., Adams, M., Russell, N., eds.: Modern Business Process Automation: YAWL and its Support Environment. Springer (2010)

[5] https://fenix-ashes.ist.utl.pt/trac/fenix-framework
[6] https://vaadin.com/home
[7] https://github.com/socialsoftware/blended-workflow

Detecting Approximate Clones in Process Model Repositories with Apromore

Chathura C. Ekanayake[1], Felix Mannhardt[2⋆], Luciano García-Bañuelos[3], Marcello La Rosa[1], Marlon Dumas[3], and Arthur H.M. ter Hofstede[1,4]

[1] Queensland University of Technology, Australia
[2] Bonn-Rhein-Sieg University of Applied Sciences, Germany
[3] University of Tartu, Estonia
[4] Eindhoven University of Technology, The Netherlands

Abstract. Approximate clone detection is the process of identifying similar process fragments in business process model collections. The tool presented in this paper can efficiently cluster approximate clones in large process model repositories. Once a repository is clustered, users can filter and browse the clusters using different filtering parameters. Our tool can also visualize clusters in the 2D space, allowing a better understanding of clusters and their member fragments. This demonstration will be useful for researchers and practitioners working on large process model repositories, where process standardization is a critical task for increasing the consistency and reducing the complexity of the repository.

1 Overview of the tool

Identification and analysis of similar process fragments, aka *approximate clones*, is a major step in business process standardization initiatives, where similar process fragments can be replaced with standardized fragments to reduce differences across different organizational units, products or brands. In order to offer concrete support to such process standardization initiatives, we developed a tool that allows analysts to identify, cluster, analyze and visualize approximate clones.

The tool is part of the Apromore advanced process model repository [5, 3]. The purpose of Apromore goes beyond that of simple model storage. Apromore aims to provide a one-stop place for the research community to expose algorithms and techniques that operate over (large) process model collections. Examples of techniques that have already been implemented are process similarity search [1] and process merging [4]. An advantage of being integrated into Apromore, is that the tool exploits Apromore's *canonical process format*, an independent format used for internal process representation. All process models imported into Apromore are converted into this internal format. Doing so, approximate clones can be detected in process models defined in different modeling languages such as BPMN, EPC, PNML, etc.

Apromore is a SaaS reachable via the Web. The functions offered by the approximate clone detection tool are available through Apromore's Web interface (the Apromore portal), as well as via Web service operations. The Apromore portal consumes

⋆ Work done while visiting Queensland University of Technology, Australia

these operations itself, but they can also be consumed by external applications (e.g. the WoPeD tool[5] – a Petri net editor – can connect to Apromore).

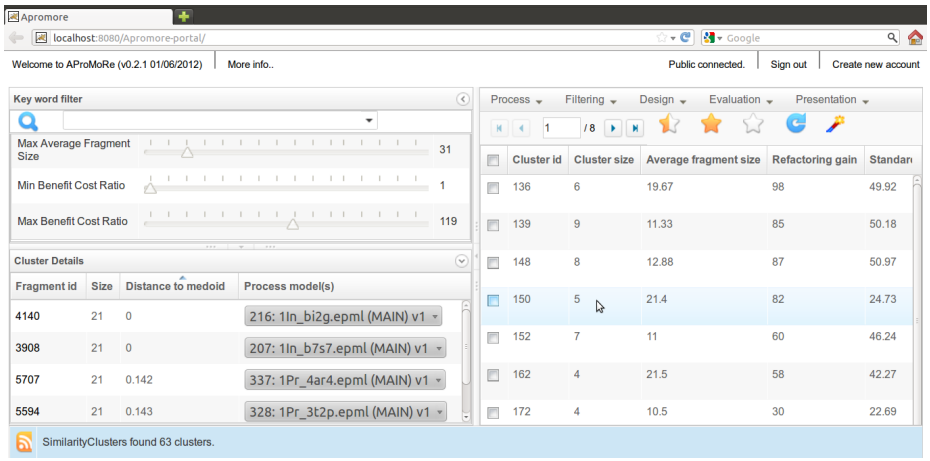

Fig. 1. Web interface of the approximate clone detection tool in Apromore

The Web interface of the approximate clone detection tool (shown in Fig. 1) provides features for creating, browsing and visualizing fragment clusters. Users can select one or more process models, specify the clustering parameters (such as the preferred clustering algorithm), and kick off the clustering. Once the fragments included in the selected process models have been clustered, users can apply different filtering criteria (e.g. on the size of the clusters) and browse the resulting clusters in a detailed list view. Another useful feature is the visualization of clusters in the 2D space. The visualization component (shown in Fig. 2) displays each fragment in a cluster as a point in the space and positions fragments within a cluster according to their distances to the medoid (distances being represented as edges between the points). It also positions the clusters in the space according to the GEDs among their medoids. One can also click on the point corresponding to a process fragment and visualize its corresponding model using any process modeling language supported by Apromore (e.g. EPCs, BPMN).

Under the hoods, the approximate clone detection tool relies on three techniques that have also been integrated into Apromore: i) RPST, ii) RPSDAG and iii) graph-edit distance. The RPST algorithm [6] is used to decompose each process model into a set of Single-Entry Single-Exit (SESE) process fragments. Such decomposed process fragments and their parent-child relationships are stored in the RPSDAG [7], an indexing structure which captures the union of the RPSTs of all process models by identifying cloned process fragments. This information about fragments and their parent-child relationships is used by a clustering algorithm to identify meaningful clusters.

[5] www.woped.org

Detecting Approximate Clones in Process Model Repositories with Apromore 3

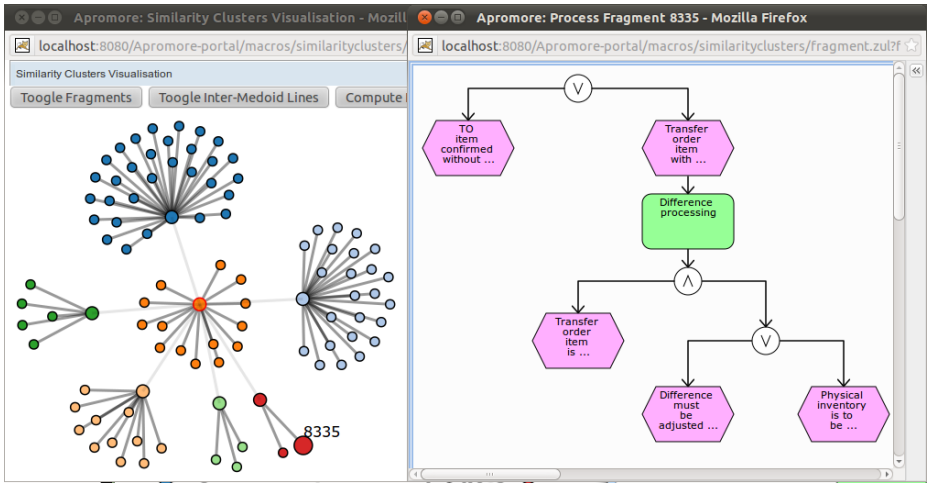

Fig. 2. Cluster visualization component of the approximate tool detection tool

Clustering algorithms need a distance measure between data objects (i.e. process fragments) in order to identify clusters. Our tool uses the graph edit distance (GED) defined in [1] as the distance measure between process fragments: this metric measures the distance between two process models (or fragments) based on a combination of structural and node labels similarity. All pairwise GEDs need be computed before a clustering algorithm can be invoked. As a process model repository can contain a large number of fragment pairs the GED calculation can be expensive. To overcome this problem, we employ several optimizations that speed up the computation. One such optimization is to exploit the RPSDAG structure to avoid the calculation of GEDs between fragments in the same hierarchy, as we do not want to have two fragments in a cluster if one fragment contains the other fragment. Once GED values are calculated, these are stored in Apromore so that users can efficiently test various clustering options using different combinations of parameters.

Clustering is performed by using data clustering algorithms. The tool features a modified version of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm and the Hierarchical Agglomerative Clustering (HAC) algorithm. However, any suitable clustering algorithm can be integrated with the tool. Once the repository has been clustered, clusters are analyzed by computing their medoid (the fragment which is the closest to the cluster center) if this is not yet available, average fragment sizes, benefit-cost ratios in terms of standardization, etc.

2 Maturity and Significance

Our tool contributes a novel method for identifying and analyzing similar fragments of possibly different process models. The approach is innovative as the application of

clustering algorithms for approximate clone detection in process models has not been studied yet. The main features of the tool are:

- Identify similar fragment groups using the HAC algorithm and a modified DBSCAN algorithm;
- Perform clustering / re-clustering operations very efficiently, in order to facilitate fine-tuning of clustering parameters;
- Analyze identified clusters to generate information useful for determining costs and benefits of standardization;
- Support filtering and browsing of clusters and their member fragments;
- Visualize clusters in the 2D space based on the GEDs between medoids and fragments;
- Visualize single fragments in various process modeling formats (e.g. BPMN, EPCs) for in-depth review.

The tool has been used to identify approximate fragment clones in two industrial process model collections: the SAP R/3 collection and a process model collection of a large insurance company under condition of anonymity. The SAP dataset contains 595 process models with sizes ranging from 5 to 119 nodes, and including 2,348 non-trivial fragments; the insurance company dataset contains 363 process models ranging from 4 to 461 nodes and including 2,037 non-trivial fragments. Both collections were clustered using our tool by using both the supported clustering algorithms and trying various clustering parameters. Once GED values were computed, the clustering phase was executed in a very short time (less than 4 seconds). Working on such short times, it was possible to experiment with different configuration parameters. The tool identified a large number of clusters (ranging from 243 to 364 clusters) from both collections using the two supported clustering algorithms. Some identified clusters contained fragments with minor differences (e.g. spelling mistakes in task labels), while some clusters contained similar fragments with more interesting differences (e.g. additional tasks, substitution of a task with a different one, additional branches, etc.). These clusters, and especially those from the latter group, could be useful for standardizing similar business processes, e.g. those that originate from copy/pasting followed by independent modifications to the copied fragments. The details of this study are available in [2].

In addition to the above studies, our tool was extensively evaluated with artificially generated datasets, to determine the accuracy of the tool in terms of precision, recall and weighted average FScore (the latter is a measure of the quality of a clustering algorithm). For this purpose, we built an evaluation framework that generates groups of similar process fragments (i.e. fragment clusters) taking content from the two industrial datasets, and integrate these fragments into separately generated process models (again, taking content from the two industrial datasets). Then those process models were given as input to our tool, which computed the clusters of approximate fragment clones and compared these with the artificially generated clusters. Average recall and precision were high, ranging from 0.71 to 0.82 (recall) and 0.84 to 0.89 (precision). The weighted average FScore was also high ranging from 0.73 to 0.77. Details of these experiments are documented in [2].

The Apromore repository can be accessed from the Apromore Web-site.[6] The source code of the approximate clone detection tool is distributed under the LGPL license along with the Apromore source code.[7]

A screencast of the tool, showcasing its main features, is available at `http://www.screenr.com/ZTn8` and `http://www.screenr.com/VTn8`.

Acknowledgments We thank all contributors to the Apromore initiative, in particular Marie-Christine Fauvet and Cameron James for their work on the implementation. This research was carried out as part of the activities of, and funded by, the Smart Services Cooperative Research Centre (CRC) through the Australian Government's CRC Programme (Department of Innovation, Industry, Science and Research).

References

1. R.M. Dijkman, M. Dumas, and L. García-Bañuelos. Graph matching algorithms for business process model similarity search. In *BPM*, volume 5701 of *LNCS*. Springer, 2009.
2. C.C. Ekanayake, M. Dumas, L. García-Bañuelos, M. La Rosa, and A.H.M. ter Hofstede. Approximate clone detection in repositories of business process models. In *BPM*. Springer, 2012.
3. M.C. Fauvet, M. La Rosa, M. Sadegh, A. Alshareef, R.M. Dijkman, L. Garcia-Banuelos H.A. Reijers, W.M.P. van der Aalst, M. Dumas, and J. Mendling. Managing Process Model Collections with APROMORE. In *Proceedings of Service-Oriented Computing (ICSOC 2010)*, volume 6470, 2010.
4. M. La Rosa, M. Dumas, R. Uba, and R.M. Dijkman. Business Process Model Merging: An Approach to Business process Consolidation. *ACM Transactions on Software Engineering and Methodology*, 2012 (to appear).
5. M. La Rosa, H.A. Reijers, W.M.P. van der Aalst, R.M. Dijkman, J. Mendling, M. Dumas, and L. García-Bañuelos. APROMORE: An Advanced Process Model Repository. *Expert Systems With Applications*, 38(6), 2011.
6. A. Polyvyanyy, J. Vanhatalo, and H. Völzer. Simplified Computation and Generalization of the Refined Process Structure Tree. In *WSFM*, 2010.
7. R. Uba, M. Dumas, L. García-Bañuelos, and M. La Rosa. Clone detection in repositories of business process models. In *BPM*, pages 248–264, 2011.

[6] `http://apromore.org`
[7] `http://code.google.com/p/apromore`

Information Flow Security for Business Process Models - just one click away

Andreas Lehmann[1] and Dirk Fahland[2]

[1] University of Rostock, Germany
andreas.lehmann@uni-rostock.de
[2] Eindhoven University of Technology, The Netherlands
d.fahland@tue.nl

Abstract. When outsourcing tasks of a business process to a third party, information flow security becomes a critical issue. In particular *implicit* information leaks are an intriguing problem. Given a business process one could ask whether the execution of a *confidential* task is kept secret to a third party which can observe some *public* (nonconfidential) tasks. A business process is secure in sense of implicit information flow if a third party can not deduce the execution of confidential tasks based on observations of public tasks. We will show that we can verify much faster whether a given process model is secure, support a new information flow property, and support the modeler to create a secure process using a graphical modeling tool. The demo might be interesting for all process modelers and those who are concerned with security in the BPM community.

1 Introduction

When outsourcing certain tasks of a business process to third-party organizations one could "leak" sensitive information (e. g., customer data, trade secrets, or financial details) to the involved third parties. This is undesirable, be it for legal or economic reasons. Information flow security concerns about such leaks which are called *interferences*, so the absence of such information leaks is called *noninterference* [6]. A standard approach to model information flow security is to label *all* tasks of a business process as either confidential or public, such a labeling is called a *complete assignment*. Given a complete assignment one could verify whether a given process is secure; however existing tools [3, 9] fail to verify industrial business processes [2]. Additionally, creating a complete assignment is cumbersome and any found interference requires a corrected and again complete assignment. We provide a solution for both problems: (1) the tool *Anica* is able to verify industrial business processes using a decomposition strategy, and (2) the modeling tool *Seda* permits a modeler to specify *partial assignments* which are automatically completed and verified by Anica. Altogether this provides modeling support and instant feedback for guaranteed noninterference.

In the rest of the paper we explain the tools *Anica* and *Seda* and report on experimental evaluation with over 550 industrial business processes [8] putting secure business processes just one click away.

2 Features

Verification. *Anica* (Automated Non-Interference Check Assistant) verifies the structural Place-based Noninterference properties PBNI+ [4] and PBNID [10] for safe Petri nets with complete assignments. Basically both, PBNI+ and PBNID, characterize noninterference violations in specific places of the process model where confidential information could be leaked to the public domain. In addition, PBNID offers additional *downgraders* (of confidentiality), which permit controlled information flow from the confidential to the public domain. This way intransitive noninterference requirements can be expressed. Although noninterference properties are defined structurally they require verification on the process behavior. Potential interferences can be identified on the net structure, the decision whether a potential interference is an active one is a verification problem on the behavior of the net; see [2, 11] for details.

Noninterference verification is not limited to Petri net-based process models. We have shown in several evaluations [2, 11] that many modeling languages, such as WS-BPEL, BPMN, and EPC can be translated to Petri nets [12], making our technique available to high-level languages as well. Also, safe nets are no restriction under the assumption of sound [1] process models (which are bounded and hence can also be represented as safe Petri nets).

To verify PBNI+ or PBNID a completely assigned and safe Petri net model of the business process is necessary. Existing tools first compute the complete state space of the net and then search for information leaks making the approach fail on large industrial business processes. Our command line interface (CLI)-based tool *Anica* instead decomposes the noninterference verification into many, typically smaller *reachability problems* [2] which can be verified using state-of-the-art model checkers using state space reduction techniques; we use *LoLA* [13]. As each noninterference violation is expressed in a specific place, those places are used to decompose the verification problem. Besides the main result *Anica* provides the following outputs: (1) colored dot files of the original assignment, (2) the found interferences together with (3) a detailed result file (certificate) and (4) a witness path (generated by *LoLA*) for each active interference. A typical industrial business process is verified in about 24 ms [2] using *Anica* and *LoLA*.

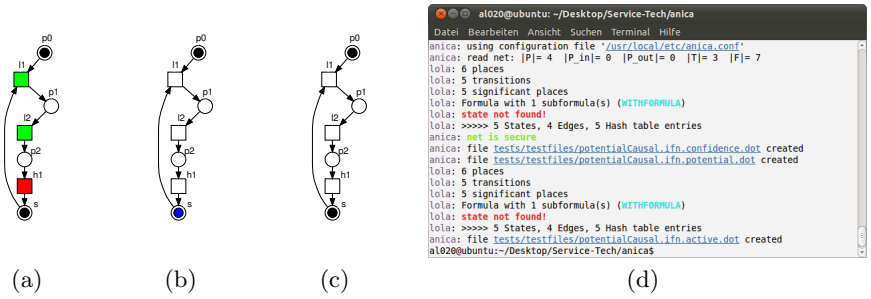

Fig. 1. Example output of Anica.

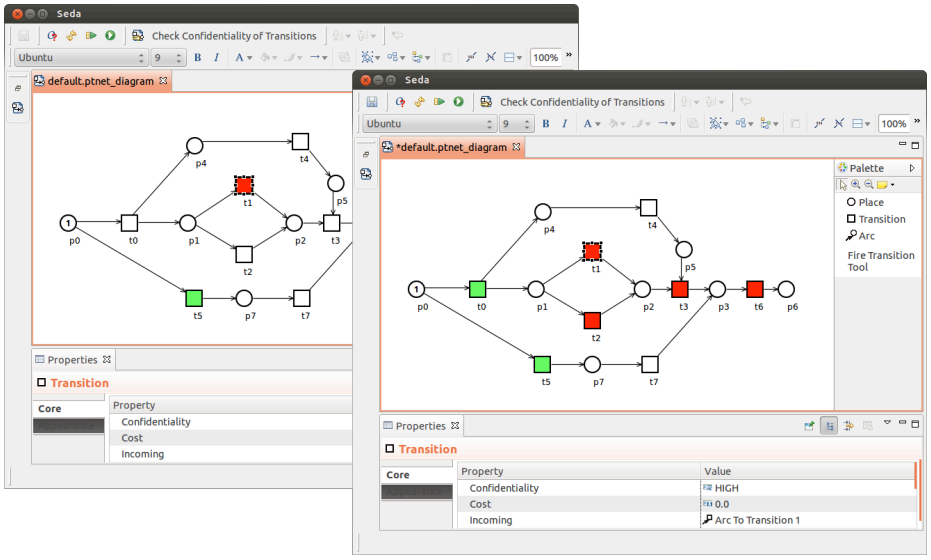

Fig. 2. Screen shots of Seda. After assigning a few transitions (left), implied assignments are calculated automatically (right).

Figure 1 shows some example dot files generated by *Anica*. The green colored tasks $l1$ and $l2$ in Fig. 1(a) are the public tasks and the red colored task $h1$ is a confidential one. As shown in Fig. 1(b) there is a potential interference in which the blue colored place s is involved (as executing $l1$ again would allow to infer that $h1$ was executed). By Fig. 1(c) this potential inferences is not active (as $l1$ can only be executed once), otherwise the place s would also be colored blue. Therefore the Petri net used in this example is secure according to the noninterference property PBNI+. A typical verification run of Anica is shown in Fig. 1(d).

Modeling support. Pure verification is often uninteresting in practical situations: a modeler would have to mark each task either as public or confidential, check interference, and if necessary reassign. This is infeasible for industrial business processes with hundreds of tasks which permit an exponential number of assignments (2^t assignments for t tasks). Rather, a modeler can create a *partial assignment* of some definitely confidential tasks and some safely public tasks, leaving other tasks *unassigned*. However, security verification requires a complete assignment.

To support the modeler in this situation, we extended our verification tool *Anica* with a so called *reasoner*, which communicates between *Anica* and the graphical editor *Seda*, an open source Eclipse-based Petri net modeling tool[3]. Seda offers the usual functionality to model and simulate Petri nets, and was extended to label each transition with a confidentiality, see Fig. 2 (left). Public tasks are labeled green, confidential tasks red, unassigned tasks are shown

[3] `http://service-technology.org/seda` (Version 1.1.3).

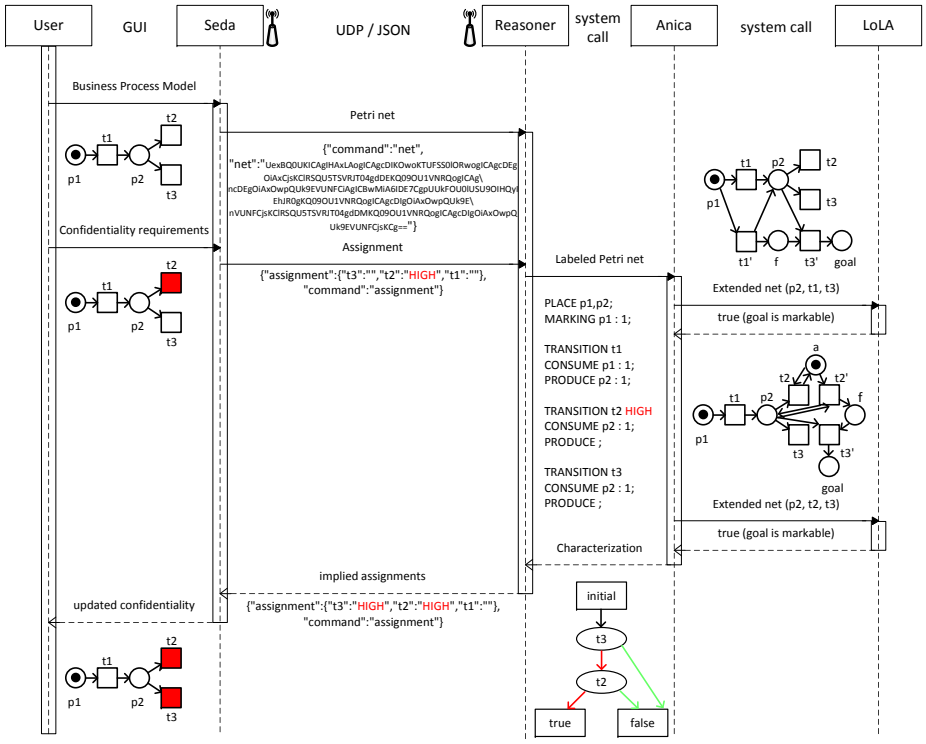

Fig. 3. Example message exchange in the current architecture.

white, i.e., in Fig. 2 (left) $t1$ is confidential, $t5$ public and the rest unassigned. The modeler may now "*Check Confidentiality of Transitions*" using a respective button which invokes the *reasoner* and *Anica*. If the current partial assignment is insecure, the modeler gets instant feedback. If the current partial assignment is secure, it is automatically extended to all other transitions that must be set to ensure the assignment chosen by the modeler, e.g., in Fig. 2 (right) four transitions were assigned to secure confidentiality of $t1$. This way the effort for the modeler is reduced and the modeler gets feedback within a second allowing for a tight integration of modeling and security verification. Still unassigned transitions can be chosen freely by the modeler. A screen cast demonstrating the modeling support is available.[4]

Architecture and implementation details. Our integration of verification in a modeling tool also has an interesting software engineering aspect. The *reasoner* acts as a middleware between the CLI-based verification tool *Anica* written in C++ and Seda written in Java based on Eclipse. In our architecture the *reasoner*, written in C++, communicates via UDP packages based on JSON which allows to use different workstations for verification and for modeling.

[4] `http://youtu.be/L7mbIHkGb7A`

Figure 3 depicts an example message exchange for a modeling session. First, the user creates a model of the business process in Seda. The model is sent to the reasoner via UDP (using Base64 encoding and JSON). After the user has assigned the confidentiality requirements (typically a partial assignment) and pressed the button "*Check Confidentiality of Transitions*", Seda sends the assignment as JSON encoded UDP package to the reasoner as well. The user expects a completed and secure assignment for the business process model. The reasoner calculates this by the help of Anica as follows. The reasoner labels the transitions of the Petri net with the given confidentiality values and hands this labeled net to Anica via a system call. For each potential noninterference violation Anica creates an extended plain Petri net, which is given to the model checker LoLA. LoLA performs a reachability check for a specific place, called "*goal*". LoLA returns true, if and only if a reachable marking (from the initial marking) exists, in which "*goal*" is marked. Based on all of LoLA's results, Anica creates a characterization of all secure assignments (encoded as a BDD) and returns it to the reasoner. From this, the reasoner infers implied assignments of currently unassigned transitions, thus deriving a secure assignment which is sent as JSON encoded UDP package to Seda. Finally, Seda colors the previously unassigned transitions according to the received message. The modeler is free to choose confidentiality values of the remaining unassigned transitions [2, 11].

3 Evaluation

Despite being a young discipline within BPM, there exist already two other tools to check PBNI+: Frau et al. implemented PNSC [9] and Accorsi et al. developed SWAT [3]. Both tools do not scale well for large Petri net models, because they require to construct and explore the complete state space of a process model suffering state space explosion. By decomposing the problem into reachability checks and by applying state-space reduction techniques, *Anica* verifies models much faster and consumes less memory [2].

We validated our technique in an experiment on verifying noninterference of industrial process models [8]. We could reduce the number of states to check from more than 30 billion to about 62,000 states. The average time consumption of 24 ms contrasts to several hours for the approach used by both other tools [2]. Additionally, we offer - to the best of our knowledge - the only tool which can verify the property PBNID as well. When used in the modeling support scenario in combination with Seda and Anica, a partial assignment could be checked and extended in about 90 ms for an average and in about 2 seconds for the largest process of [8], demonstrating its feasibility for industrial business processes [11]. Furthermore *Anica*, the *reasoner* and *Seda* are publicly available[5] in contrast to PNSC [9] and SWAT [3].

[5] `http://service-technology.org/anica`

4 Conclusion

Lessons learnt. Our approach to decompose a verification problem into many typically smaller problems makes the verification of noninterference applicable for industrial business processes. Furthermore the achieved speed allows a model support based on partial assignment which are typically unusable for pure verification tasks and tools, but as much more interesting for practical domains.

Future work. The next step could be the integration of the noninterference checks in the context of other business process modeling tools (e.g, Oryx [5]) to support BPMN rather than Petri nets. This would require a new reasoner to communicate between *Anica* and for instance Oryx together with a background translation between BPMN and Petri nets [7]. Moreover the current error message in case of an insecure assignment can be extended to a detailed diagnostic information. Therefore the witness path (already provided by Anica and LoLA) could be visualized in Seda using its simulation feature.

Acknowledgement. The authors cordially thank Niels Lohmann for his ideas about the reasoner. This work was partially funded by the German Research Foundation in the project WS4Dsec in the priority program Reliably Secure Software Systems (SPP 1496).

References

1. Aalst, W.M.P.v.d.: The application of Petri nets to workflow management. Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
2. Accorsi, R., Lehmann, A.: Automatic information flow analysis of business process models. In: BPM 2012. pp. 172–187. LNCS 7481, Springer (2012)
3. Accorsi, R., Wonnemann, C., Dochow, S.: SWAT: A security workflow toolkit for reliably secure process-aware information systems. In: ARES 2011. pp. 692–697. IEEE (2011)
4. Busi, N., Gorrieri, R.: Structural non-interference in elementary and trace nets. Mathematical Structures in Computer Science 19(6), 1065–1090 (2009)
5. Decker, G., Overdick, H., Weske, M.: Oryx - an open modeling platform for the bpm community. In: BPM 2008. pp. 382–385. LNCS 5240, Springer (2008)
6. Denning, D.E., Denning, P.J.: Certification of programs for secure information flow. Commun. ACM 20(7), 504–513 (1977)
7. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in bpmn. Information & Software Technology 50(12), 1281–1294 (2008)
8. Fahland, D., Favre, C., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Analysis on demand: Instantaneous soundness checking of industrial business process models. Data Knowl. Eng. 70(5), 448–466 (2011)
9. Frau, S., Gorrieri, R., Ferigato, C.: Petri net security checker: Structural non-interference at work. In: FAST 2008. pp. 210–225. LNCS 5491, Springer (2008)
10. Gorrieri, R., Vernali, M.: Foundations of security analysis and design vi. chap. On intransitive non-interference in some models of concurrency, pp. 125–151. Springer (2011)
11. Lehmann, A., Lohmann, N.: Modeling wizard for confidential business processes. In: SBP 2012. Tallinn, Estonia (2012)
12. Lohmann, N., Verbeek, H., Dijkman, R.M.: Petri net transformations for business processes – a survey. LNCS ToPNoC II(5460), 46–63 (2009)
13. Wolf, K.: Generating Petri net state spaces. In: Kleijn, J., Yakovlev, A. (eds.) ICATPN 2007. LNCS 4546, vol. 4546, pp. 29–42. Springer-Verlag (2007)

Disco: Discover Your Processes

Christian W. Günther and Anne Rozinat

Fluxicon
Bomanshof 259, 5611 NS, Eindhoven, The Netherlands.
{christian,anne}@fluxicon.com

Abstract. Disco is a complete process mining toolkit from Fluxicon that makes process mining fast, easy, and simply fun.

1 Why Disco?

As former process mining researchers, we started Fluxicon in 2009 to build professional tools that help organizations to regain control over their processes.

Our first product Nitro addressed the pain of getting the original process data from IT systems into a format that can be used for process mining. Today, Nitro is used all over the world by practitioners and researchers to convert raw data into event logs that can be analyzed with the leading academic process mining toolkit ProM.

While ProM is great and immensely powerful, we realized through our own process mining consulting projects, and through many conversations with practitioners, that process analysts in practice need a tool that—above all—makes process mining easy and fast. And this is what Disco is all about.

2 Tour

The following tour gives you an overview about the main functionality of Disco.

2.1 Import

Every process mining project starts with the data that should be analyzed. Disco has been designed to make the data import really easy by automatically detecting timestamps, remembering your configuration settings, and by loading data sets with high speed.

One simply opens a CSV or Excel file and configures which columns hold the case ID, timestamps, activity names, which other attributes should be included in the analysis, and the import can be started. Data sets are imported in a read-only mode, so the original files cannot be modified (which is important, e.g., for auditors).

Disco is also fully compatible with the academic toolsets ProM 5 and ProM 6. By importing and exporting the event log standard formats MXML and XES, advanced users can seamlessly move back and forth between Disco and ProM if they want to benefit from the new research technologies developed in academia.

Disco also features a short-cut import and data exchange for previously imported data sets with up to 200x speed-up for very large data sets through the native FXL Disco log file format.

2.2 Automated Process Discovery

The core functionality of process mining is the automated discovery of process maps by interpreting the sequences of activities in the imported log file. After one presses the Start import button the user is taken right into the Map view, where she can quickly and objectively see how the process has been actually performed.

Disco uses an intuitively understandable and 100% truthful process map visualization. The thickness of paths and coloring of activities show the main paths of the process flows, and wasteful rework loops are quickly discovered.

The Disco miner is based on the Fuzzy miner, but has been further developed in many ways. The Fuzzy Miner was the first mining algorithm to introduce the "map metaphor" to process mining, including advanced features like seamless process simplification and highlighting of frequent activities and paths. For Disco, we have used the approach of the Fuzzy Miner and combined it with experience from our own practice and user testing.

The result is a mining algorithm that, while providing reliable and trustworthy results for data sets of arbitrary complexity, can be operated and understood efficiently by domain experts with no prior experience in process mining. Although the Disco miner is based on the framework of the Fuzzy Miner, we have developed a completely new set of process metrics and modeling strategies, effectively making the Disco miner a next-generation Fuzzy Miner.

Our design priorities are what sets the Disco miner apart from other solutions:

1. *Usability*: Our goal was to have a miner that can be operated and understood by domain experts, with an adequate learning curve to also accommodate process mining experts. We also have put great effort into making our visualizations information-dense, while avoiding information overload. For Disco, we have used state-of-the-art UX and visualization research, user testing, and lots of development time to make sure our models are nice to read and quick to understand.
2. *Fidelity*: Creating a truthful model from a simple, well-structured process model is easy. When faced with complex data, though, most commercial approaches resort to drastically limiting the data used (only using the mainstream variants) to keep model complexity in check. We wanted a miner that can intelligently extract the most important parts of the process from the full set of data, and create a useful process model from data of arbitrary complexity.
3. *Performance*: Almost all process mining tools want to be used in a procedural fashion: You give them the data, and some parameters, they create a process model, done. We see process mining as an explorative and highly interactive task, where the domain expert learns to understand the data by looking at the process from multiple perspectives in quick succession. For this approach to work, we need our miner to work very fast.

The Disco miner is considerably faster than any other approaches we are aware of, while delivering superior model quality. We think there is inherent value in having a good approximation of complex behavior in a few seconds, versus a perfect model in three hours (which is what you get with, e.g., genetic approaches). By intensively optimizing the whole stack, down from the log storage layer up to the graph visualization,

we have created a miner that fosters truly interactive usage which, ultimately, leads to better and more meaningful analysis results.

2.3 Process Statistics

Next to the process maps one can also inspect statistics about the process. For this, one simply changes to the Statistics tab in the toolbar. The user will get overview information about the number of cases and events in the data set, the time frame covered, and performance charts like, for example, about the case duration.

Further statistics views provide frequency and performance information for all activities and resources in the process. Furthermore, there are statistics for any additional data attribute column that was included in the data set. These additional data attributes are usually very important for the process analysis, because they hold relevant context information such as:

- Which *product* a service call was about,
- Which type of *category* a change request in an IT Service process falls in,
- The *channel* through which a lead in a sales process came in,
- *Domain-specific characteristics* such as warranty vs. out-of-warranty repairs in a service process,
- By which *department* the activity was handled,
- In which *country* the process was performed,
- The *value* of an order, which is relevant for many purchasing processes, because depending on the amount of money that is involved different anti-fraud rules will apply, etc.

In our projects, we often get data sets with up to 40 or 60 additional data attributes that are relevant and can be used in the analysis. Disco shows the users these attribute statistics, but also lets them use them to drill down and focus their analysis, and to split out and compare processes with respect to these categories.

2.4 Variants and Individual Cases

The third data set view is the Cases tab. While the Map view gives an understanding about the process flows, and the Statistics view provides detailed performance metrics about the process, the Cases view actually goes down to the individual case level and shows the raw data.

To be able to inspect individual cases is important, because one will need to verify the findings and see concrete examples particularly for “strange” behavior that will most likely be discovered in the process analysis. Almost always users find things that are hard to believe until they have drilled down to an individual example case, noted down the case number, and verified that this is indeed what happened in the operational system.

Furthermore, looking at individual cases with their history and all their attributes can give additional context (like a comment field) that sometimes explains why something happened. Finally, being able to drill down to individual cases is important to be able to

act on the analysis. For example, if one has found deviations from the described process, or violations of an important business rule, one may want to get a list of these cases and talk to the people involved in them to provide additional training.

In addition to a complete list of all cases in the data set, the user also gets direct access to the variants in the process. Variants are an integral part of the process analysis. In Disco, a variant is a specific sequence of activities. It can be seen as one path from the beginning to the very end of the process. In the process map, an overview of the process flow between activities is shown for all cases together. A variant is then one "run" through this process from the start to the stop symbol, where also loops are unfolded. Usually, a large portion of cases in the data set are following just a few variants, and it is useful to know which are the most frequent ones.

Furthermore, a live full text search across case names and all activity, resource, and data columns lets the user find specific cases based on the words or word fragments she is looking for.

2.5 Filtering

Disco offers powerful, non-destructive filtering capabilities for explorative drill-down, and for focusing the analysis. These filters are quickly accessible from any view and easy to configure.

In total, there are six powerful filter types available in Disco, and they can be combined and stacked in any order:

- The *Timeframe filter* with intuitive calendar controls to select cases and events based on a time window. It can be used, for example, to compare the processes before and after a process change.
- The *Variation filter* that allows one to focus the analysis on either the mainstream behavior or precisely the exceptional cases by making use of the variants from the Cases view.
- The *Performance filter* to focus on cases based on a variety of different performance metrics like, for example, the case duration.
- The *Endpoints filter* to select cases based on their start and end activities. For example, one can filter incomplete cases, or trim cases to cut out a part of the process.
- The *Attribute filter* to focus on (or exclude) certain activities, resources or process categories based on data attributes.
- The *Follower filter* for powerful process pattern-oriented filtering, including a 4-Eyes filter option that can be used to check for segregation of duty violations.

Together with the three analysis views, these filtering capabilities enable Disco users to quickly and interactively explore their process into multiple directions, and to answer concrete questions about the process. Because filtering, and Disco in general, are so fast, one can also hold interactive process workshops, where the analyst and a group of other process stakeholders get together to do an As-Is analysis and generate process improvement ideas along the way.

2.6 Performance Highlighting

In addition to the frequency-based process map, one can also analyze the time that is spent in the process. The average durations of the activities and the inactive (waiting) times between activities are automatically extracted from the timestamps in the data set and visually projected onto the process map.

An alternative Total durations performance highlighting option shows these high-impact areas at one glance by summing up the durations for each activity and path for the complete data set.

2.7 Animation

Animation is a way to visualize the process flow over time right in the discovered process map (a bit like showing a "movie" of the process). Animation should not be confused with simulation. Rather than simulating, the real events from the log are replayed in the discovered process map as they took place.

Animation can be very useful to communicate analysis results to process managers or other people who are no process analysis experts. By showing how the cases in the data set move through the process (at their relative, actual speed), the process is literally "brought to life".

2.8 Project Management

One of the advantages of Disco is that it supports project work through the management of multiple data sets in one project view. In a typical process mining project, one will import log files in different ways, filter them, and make copies to save intermediate results. This results in many different versions and views of the data sets and can easily get out of hand.

The project view in Disco is there to help the users keep an overview. It keeps all their work in one place and lets them make notes about what they found out, or what they still want to check. Complete projects can be exported and shared with other people who can start right where they left off.

Disco features a sandbox project that we prepared for new users to get started quickly after the installation of Disco.

3 Links

A 6-min screencast has been recorded for this demo. You can watch this screencast in two parts, Part I at http://screenr.com/F1n8, and Part II at http://screenr.com/q1n8.

Furthermore, you can view the Disco product page and download a free demo version at http://fluxicon.com/disco/. You can also read a tour including screenshots and examples in our launch blog post here: http://fluxicon.com/blog/2012/05/say-hello-to-disco/.

Note that we provide free academic licenses for Disco in our Academic Initiative for Process Mining Research and Education (see http://fluxicon.com/academic/).

Mayflower—Explorative Modeling of Scientific Workflows with BPEL

Mirko Sonntag, Michael Hahn and Dimka Karastoyanova

Institute of Architecture of Application Systems, University of Stuttgart,
Universitaetsstrasse 38, 70569 Stuttgart, Germany
{sonntag, hahn, karastoyanova}@iaas.uni-stuttgart.de

Abstract. Using workflows for scientific calculations, experiments and simulations has been a success story in many cases. Unfortunately, most of the existing scientific workflow systems implement proprietary, non-standardized workflow languages, not taking advantage of the achievements of the conventional business workflow technology. It is only natural to combine these two research branches in order to harness the strengths of both. In this demonstration, we present Mayflower, a workflow environment that enables scientists to model workflows on the fly using extended business workflow technology. It supports the typical trial-and-error approach scientists follow when developing their experiments, computations or simulations and provides scientists with all crucial characteristics of the workflow technology. Additionally, beneficial to the business stakeholders, Mayflower brings additional simplification in workflow development and debugging.

Keywords: Scientific workflows, Model-as-you-go, SOA, BPEL.

1 Scientific Workflows

The introduction of workflows to scientific computations and simulations has proven beneficial for scientists in many domains, e.g. image processing in physical astronomy [1], earthquake simulations in geology [2], or simulation regarding the biodiversity of species [3]. Workflows speedup scientific computations through automation and straightforward parallelization of tasks, reduce the programming effort for scientists, and improve traceability of scientific results. There is a broad spectrum of scientific workflow systems available, such as Kepler[1], Triana[2], Taverna[3] and Pegasus[4]. Most of these systems have been developed from scratch, implement proprietary, non-standardized workflow languages, and serve specific scientific application domains.

There are also approaches to enhance tools and concepts of the business workflow technology in order to facilitate modeling and execution of scientific computations

[1] https://kepler-project.org/
[2] http://www.trianacode.org/
[3] http://www.taverna.org.uk/
[4] http://pegasus.isi.edu/

and experiments [4, 5, 6]. We are convinced that the well-established conventional workflow technology brings many advantages compared to existing solutions for scientific workflows, namely: (1) the technology is generic and thus independent of the scientific domain and can be applied to almost every scenario, (2) the concept of workflow models and instances can be used to conduct scientific parameter sweeps and parallelize computations, (3) the human tasks features are helpful to integrate human decision points and steering, (4) existing concepts for adaptation of workflows increase the flexibility of scientific workflows, and (5) conventional workflows are standard-based, which facilitates collaboration between scientists and reuse.

The life cycle of business workflows differs from that of their scientific counterpart [4]. That is one of the reasons why the current business workflow technology needs profound extension to be applicable in the scientific domain. Scientists develop software and workflows much more explorative and experimental in a trial-and-error manner [4, 6, 7]. They know the goal of their research but often not the exact way towards this goal or the exact (intermediary) results. That means the scientists approach their goals by trying out different parameter values, by adding or removing activities, by repeating steps of an experiment, or by using different solvers for equations. From the viewpoint of the conventional workflow technology the workflow modeling and runtime phases are not strictly separated, but they are experienced by scientists as a single phase because they can alternate arbitrarily. Furthermore, the workflows are not deployed by scientists. Instead, they are simply started and the workflow deployment is hidden behind a *run* operation. The scientists are the driver of all life cycle phases. They need a single, integrated, easy-to-use tool to deal with workflows in their respective phases without switching between the tools.

We call this flexible development of scientific workflows *Model-as-you-go* since workflows are modeled on the fly during execution [4]. The approach simplifies workflow modeling and increases the robustness of workflows because the user can fix structural failures, repair the workflow context or handle runtime faults without restarting the workflow. Model-as-you-go also integrates approaches for workflow flexibility such as ad hoc adaptations, instance migration, versioning and ad hoc backward loops. The challenges are (1) the start and configuration of workflows via a simple *run* operation instead of a full-blown deployment mechanism; (2) the correlation between processes in the engine and the modeling tool; (3) the combination of a process modeling tool and an instance monitor as entities originally designed for different life cycle phases; (4) the management of process models and instances in the modeling/monitoring tool; (5) the impact of semantically dependent activities on the adaptation of processes; and (6) the stepwise execution of workflows using a non-intrusive event model [8].

In this demo, we present an implementation of the Model-as-you-go concepts: *Mayflower*, the Model-as-you-go Workflow Developer. Mayflower uses BPEL as workflow language and is built upon existing BPEL implementations, namely the Eclipse BPEL Designer[5] as modeling tool and the Apache ODE[6] as workflow engine.

[5] http://eclipse.org/bpel/

[6] http://ode.apache.org/

The software makes development and debugging of BPEL workflows easier and hence can also be used in business scenarios.

2 Architecture Walkthrough

Mayflower consists of five major parts (Fig. 1): (1) The scientist interacts with the Eclipse-based *Modeling Framework*. It consists of the Eclipse BPEL Designer as workflow editor. We have extended the BPEL Designer with functions to (a) control workflow execution (run/resume, suspend and terminate), (b) monitor workflow instances using state information of the workflow engine, (c) adapt the logic and functions dimensions of running workflows as well as the workflow context (e.g. the content of variables, activity markings in order to enforce backward loops), (d) manage different versions of workflow models, (e) specify breakpoints, and (f) track changes made by the user to fill the provenance record. Further, there are plug-ins to access and administrate the Resource Manager and the Workflow Engine.

(2) The *Workflow Engine*, an extended Apache ODE, (a) instantiates workflow models, navigates through the workflow graphs and (b) invokes scientific computations exposed as Web services. It also contains components to (c) handle process models and instances (e.g. resume, suspend, terminate), (d) deploy and undeploy process models, (e) adapt the logic of running workflows, (f) publish execution events, and (g) access and modify the context of workflow instances.

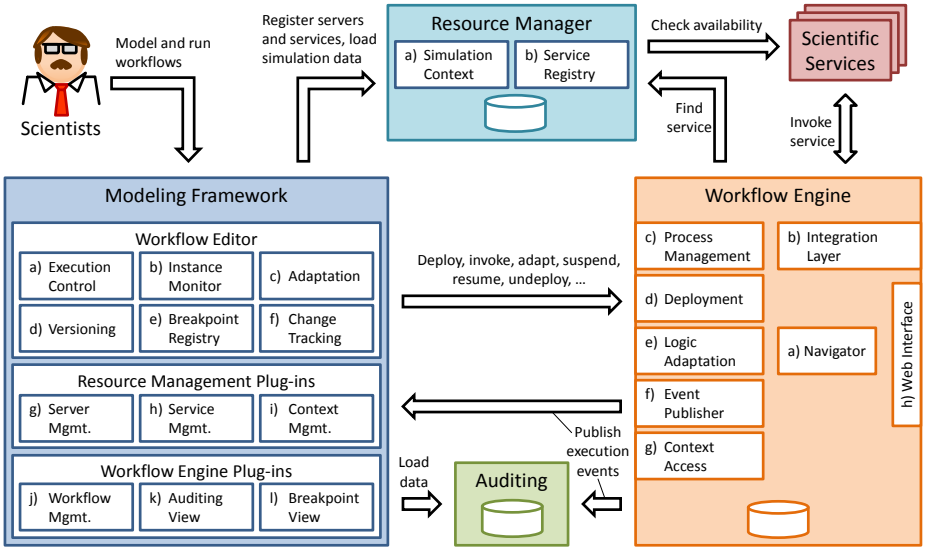

Fig. 1. Overview of the Architecture.

(3) The *Auditing* component stores execution events for workflow instances published by the engine. This information allows loading the state of all workflow instances into the instance monitor of the workflow editor, even of those that were not

started by the editor. It also correlates the workflow model in the engine with that in the modeling framework, since they have different representations and identifiers. (4) The *Resource Manager* (a) offers a logically centralized storage for simulation data and (b) works as registry for the scientific services that participate in the simulations and calculations. (5) The *Scientific Services* provide the domain-specific logic that is orchestrated by the scientific workflows running on the workflow engine.

Fig. 2 shows the user interface of Mayflower and some of its components:

1. Workflows can be started, suspended, resumed and terminated via a toolbar extension.
2. The instance monitor is an extension of the editor pane. Workflow models are enriched with instance information and the activities are colored according to their execution state (see the legend in Fig. 2).
3. The process instance state is displayed in the upper left corner of the editor pane.
4. Breakpoints can be specified in the properties of activities via the new *debug* tab. One or more execution events of an activity can be registered as breakpoint.
5. A reached breakpoint is signaled via a highlighted activity.
6. The user can skip the breakpoint with the help of a toolbar function.

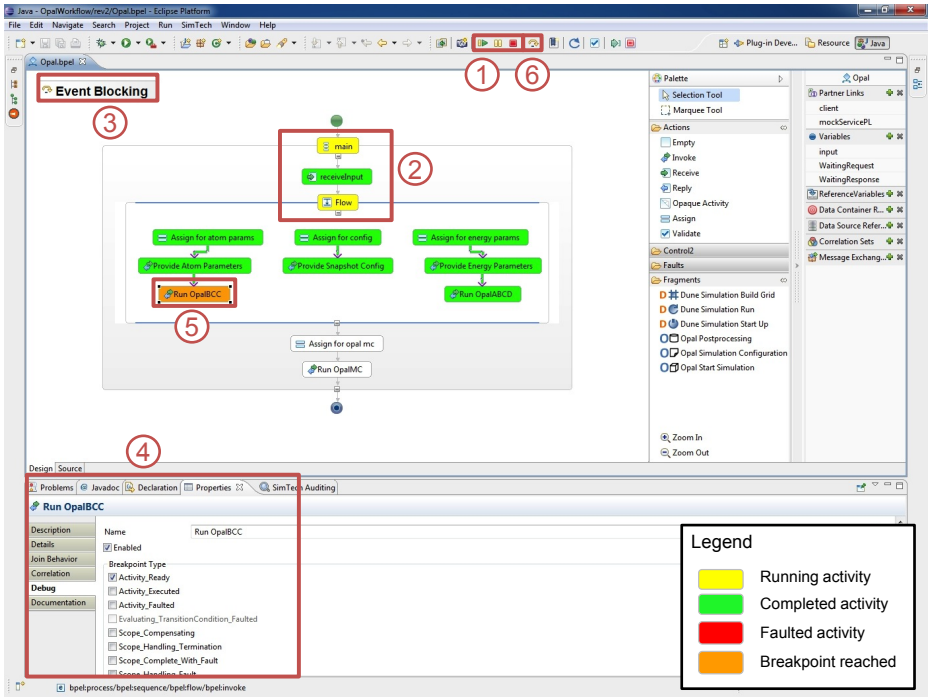

Fig. 2. Screenshot of Mayflower's Modeling Framework with its main components.

3 Demonstrated Features

We demonstrate Mayflower with the help of a use case for the simulation of the ageing process in copper-alloyed steel, an example of a solid-body simulation. The simulation computes how the atomic structure of steel changes when being operated over many years. The atoms exchange their positions and build precipitations or clusters that negatively influence the material properties (Fig. 3). We have re-engineered the simulation tool with the help of BPEL and Web services in order to speed up the simulation runtime through automation of formerly manual tasks and parallelizing post-processing steps [9]. The workflow will be the basis for the demonstration.

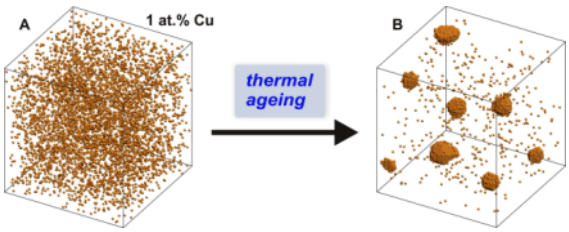

Fig. 3. Simulation of the ageing process in copper-alloyed steel [10]

The following aspects of Mayflower are shown:

1. *Workflow modeling, execution and monitoring.* The user can drag activities from the palette and drop them on the editor pane in order to specify the logic of the workflow. When a valid workflow model (fragment) is created, the user starts the workflow and puts in parameter values he is requested for by the tool. Monitoring of the running workflow starts automatically; the state is displayed by changing the activities' color based on workflow execution events.
2. *Usage of breakpoints.* Breakpoints are specified in the properties of activities. Workflow execution is paused when a breakpoint is reached. It is possible in parallel branches to pause one path while the other path continues.
3. *Adaptation and Versioning.* Modification of running workflows is experienced by scientists as workflow modeling: activities can be added, removed, or changed, while the process instance is already being executed. The changes are propagated to the engine with versioning and instance migration techniques. Furthermore, the user can inspect and change variable values and conduct ad hoc backward loops.
4. *Ad hoc rerun of activities.* The user can jump backwards to an arbitrary activity in the past of the workflow instance. He can select to compensate already completed work in the iteration body. As input for the next run of the activities it is possible to take either the current variable values or variable values that were valid at a former time step during workflow execution [11].
5. *Resource Management.* Via the resource manager plug-in the user provides the servers and scientific services that should be taken for a simulation. He can monitor the workload in the system by having a look at the number of service requests and the number of granted service usage tickets. Furthermore, the user can inspect intermediary results in the simulation context [9].

4 Maturity of the Software and Screencast

The software is a prototypical implementation of the Model-as-you-go concept developed in the past three years in the scope of our work in the research cluster SimTech[7]. A demonstration video of the tool is available online[8].

Acknowledgements. The authors would like to thank the German Research Foundation (DFG) for financial support of the project within the Cluster of Excellence in Simulation Technology (EXC 310/1) at the University of Stuttgart.

References

1. G. B. Berriman, E. Deelman, J. Good et al.: Generating Complex Astronomy Workflows. In: I. Taylor, E. Deelman, D. B. Gannon, M. Shields (Eds.): Workflows for e-Science. Springer, 2007
2. P. Maechling, E. Deelman, L. Zhao et al.: SCEC CyberShake Workflows—Automating Probabilistic Seismic Hazard Analysis Calculations. In: I. Taylor, E. Deelman, D. B. Gannon, M. Shields (Eds.): Workflows for e-Science. Springer, 2007
3. A. Jones: Workflow and Biodiversity e-Science. In: I. Taylor, E. Deelman, D. B. Gannon, M. Shields (Eds.): Workflows for e-Science. Springer, 2007
4. M. Sonntag and D. Karastoyanova: Next Generation Scientific Experimenting Based On the Workflow Technology. Proceedings of the 21st IASTED International Conference on Modeling and Simulation (MS'10), 2010
5. A. Akram, D. Meredith, and R. Allan: Evaluation of BPEL to scientific workflows. In: CCGRID '06: Proc. of the 6th IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society, 2006, pp. 269–274
6. I. Wassink, M. Ooms, and P. van der Vet: Designing workflows on the fly using e-BioFlow. Lecture Notes in Computer Science (LNCS), vol. 5900, 2009, pp. 470–484
7. G. Vossen and M. Weske: The WASA approach to workflow management for scientific applications. In: Dogac et al. (Eds.): Workflow Management Systems and Interoperability, NATO ASI Series F: Computer and System Sciences, vol. 164, Springer-Verlag, Berlin, 1998, pp. 145–164
8. O. Kopp, S. Henke, D. Karastoyanova et al.: An event model for WS-BPEL 2.0. Technical Report No. 2011/07, University of Stuttgart, Germany, 2011
9. M. Sonntag, S. Hotta, D. Karastoyanova, D. Molnar, and S. Schmauder: Using services and service compositions to enable the distributed execution of legacy simulation applications. In: Proceedings of the 4th European Conference ServiceWave 2011, Poznan, Poland, 2011.
10. D. Molnar, P. Binkele, S. Hocker, and S. Schmauder: Multiscale modelling of nano tensile tests for different Cu-precipitation states in α-Fe. In: Proceedings of the 5th International Conference on Multiscale Materials Modelling, Fraunhofer Verlag, 2010, pp. 235–239
11. M. Sonntag and D. Karastoyanova: Ad hoc Iteration and Re-execution of Activities in Workflows. In: IARIA (Hrsg): International Journal On Advances in Software, ISSN 1942-2628, vol. 5, no. 1 & 2, Xpert Publishing Services, 2012, pp. 91–109

---

[7] http://www.simtech.uni-stuttgart.de/index.en.html

[8] http://www.iaas.uni-stuttgart.de/institut/mitarbeiter/sonntag/indexE.php#mayflowerVideo

CRISTAL: Collection of Resource-centrIc Supporting Tools And Languages*

Cristina Cabanillas, Adela del-Río-Ortega, Manuel Resinas, and Antonio Ruiz-Cortés

Universidad de Sevilla, Spain
{cristinacabanillas, adeladelrio, resinas, aruiz}@us.es

Abstract. In this demo, we introduce CRISTAL (Collection of Resource-centrIc Supporting Tools And Languages), a tool suite aimed at improving the human resource management capabilities of current Business Process Management Systems (BPMSs), covering the design and enactment phases of the business process (BP) life cycle. The central element is Resource Assignment Language (RAL), a Domain Specific Language (DSL) for specifying resource assignments in process models. RAL's strong analysis capabilities enable the automated resolution of resource assignment expressions both (i) at design time, serving for post-design analysis to find and correct potential problems prior to execution, and (ii) at run time, in order to execute the BP in an existing BPMS considering the RAL assignments for resource allocation. The resource assignments can be directly modelled in a Business Process Modelling Notation (BPMN) diagram, or specified by means of a RACI matrix. In the latter case, CRISTAL can take all the RACI information automatically and introduce it into a resource-unaware BPMN model at any moment, resulting in a RACI-aware BP model (and, thus, a resource-aware BP model).

1 Background. RAL and RACI matrices

RAL is a DSL specifically developed to express resource assignments for the activities of a BP [1]. The language was designed to bridge the gap between BP models and organizational models, and to exceed the scope of existing approaches. RAL expressions cover from simple assignments based on specific members of the company, to complex assignments containing access-control constraints (e.g. Segregation of Duty -SoD-) between activities, as well as compound expressions. As can be seen in the following examples, its syntax is close to natural language, which increases its understandability:

```
RAL 1: IS Samuel
RAL 2: NOT (IS PERSON WHO DID ACTIVITY CreateResolutionProposal)
RAL 3: (HAS ROLE DocumentWriter) OR (HAS POSITION ACDocumentSigner)
```

* This work has been partially supported by the European Commission (FEDER), Spanish Government under project SETI (TIN2009-07366); and projects THEOS (TIC-5906) and ISABEL (TIC-2533) funded by the Andalusian Local Government.

We provided RAL with formal semantics based on Description Logics (DLs) in order to be able to automatically solve RAL expressions. It also enabled us to benefit from operations implemented in DL reasoners to analyse BPs in terms of how resources are being managed. RAL's semantics now cover both the design time [2] and the run time phases of the BP life cycle [3], meaning that:

- RAL assignments can be automatically solved at design time. Accuracy is not possible for those expressions requiring run-time information though, since some data are missing (e.g. the expressions related to the person who did a certain activity). Besides, the design-time resource-related analysis of RAL expressions automates the answering of questions such as (i) who are the potential performers of each BP activity?; or (ii) what is the potential set of activities each person of an organization can be allocated at run time?
- Regarding run time, resource assignments defined with RAL can be automatically solved during execution to obtain the *real* potential performers of an activity given the specific run-time information. The allocation method then depends on the support provided by the Business Process Management System (BPMS) in which the process is executed.

As explained in [1], RAL can be easily used in conjunction with BPMN 2.0 because it can be directly integrated into it by using the resource assignment mechanisms provided by the standard [4]. Nonetheless, it could also be integrated into other workflow (WF) modelling notations, provided that they offered a flexible way to define resource assignments in a BP model.

Furthermore, RAL can also be used together with RACI matrices to specify resource assignments in a BP model. RACI matrices constitute an alternative for enhancing the management of human resources in an organization, providing detailed information about the degree of involvement of the members of the company in the activities carried out within it [5]. This degree of involvement is specified by means of the so-called RACI roles, which usually are: Responsible (R), Accountable (A), Consulted (C) and Informed (I). This extends the traditional notion of resource assignments in BP models which only defines the resource that is in charge of the activity.

2 CRISTAL's Overview. Demonstration Guideline

CRISTAL is composed of several tools, to be named RACI2BPMN, DT RAL Solver and RT RAL Solver, which facilitate the definition and analysis of resources in BP models. These tools can be used separately or sequentially. Figure 1 provides an overview of the system, in which the tools are represented in rounded rectangles, their inputs and outputs are represented by documents linked with dashed arrows, and the possible interconnection between tools is done by using solid arrows. One possible way to use CRISTAL's tools is the following.

We can use Oryx [6] or any other process editor supporting BPMN to build a resource-unaware BPMN model, i.e. a process model that does not contain

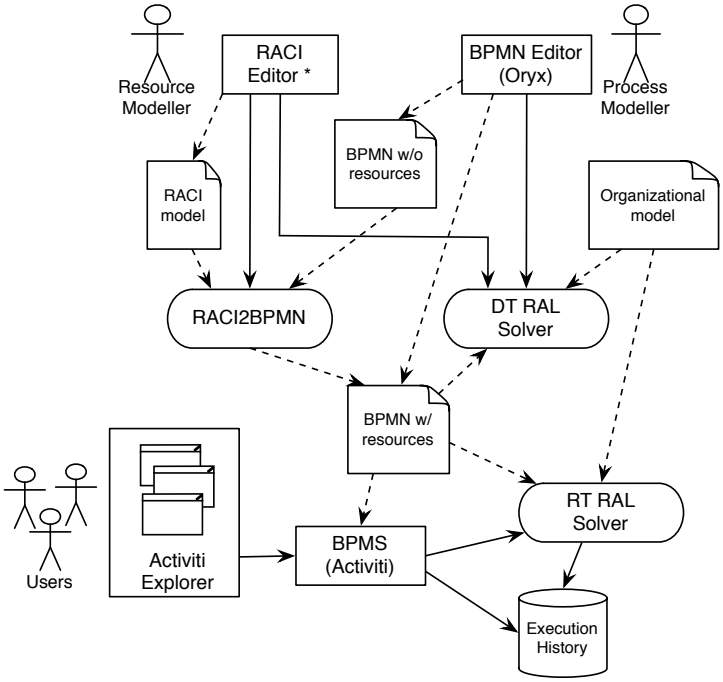

Fig. 1: Overview of CRISTAL

any resource assignments. Then, with CRISTAL's RACI Editor we can build a RACI matrix with the responsibility assignments we aim to apply to the process activities. In order to generate accurate resource assignments for the process, the RACI matrix may need to be extended with some *binding information*. CRISTAL's RACI Editor is a Web application that allows the definition of both the RACI matrix and the binding information. It is currently under development.

With these inputs, the RACI2BPMN tool can automatically turn the resource-unaware process model into a RACI-aware BPMN model, i.e. it contains the assignments from the RACI matrix and the binding information. The resource-related information in the model is expressed with RAL. The resulting BP model can be opened in any editor that supports BPMN 2.0 (e.g. Oryx), so both the process model and the resource assignments can be modified as desired.

The next tool to be executed may be DT RAL Solver, which allows us to automatically analyse the RAL expressions associated to the activities of a RAL-aware BP model. Among other operations, this tool automatically calculates the set of potential performers for each activity in the process model. To do so, the tool must receive the OWL description of an organizational model together with the RAL-aware BP model.

Finally, RAL-aware BP models can also be opened in an open-source BPMS called Activiti[1], in which we have previously integrated the RT RAL Solver tool. The goal of this tool is to automatically solve the RAL expressions at run time. Then, the proper activities are automatically offered to the potential performers resulting from the assignments at run time, to proceed with the allocation of the tasks to specific resources. The execution history is stored in execution logs.

The low-level description of CRISTAL's architecture can be found at `www.isa.us.es/cristal/`.

3 Maturity and Significance to the BPM Field

The DT RAL Solver tool was developed in 2011, as a prototype to demonstrate how DL reasoners could be used to automatically solve RAL expressions at design time [2]. Driven by research results, it was then extended to be provided with the proper implementation to solve them considering run-time data as well. In the current version, RT RAL Solver is still a prototype.

The RACI2BPMN tool has been recently developed from research results on how to combine BPs and RACI matrices, given that the resource-related information provided by the matrices is greater than the information that today's BP modelling notations allow to specify [7]. The main features currently provided by CRISTAL are the following:

1. **Automated generation of a BP model with complete information about the RACI roles involved in the BP.** The model can be opened and modified in any modelling tool supporting BPMN, since it is BPMN-compliant. In addition, it is prepared to be directly executed in a BPMS.
2. **Design-time automated resolution of all the RAL expressions defined in [1].** As a result, the potential performers of the activities of a BP model can be automatically inferred from their RAL expressions at design time. The design-time features are available as a plug-in for Oryx [6].
3. **Run-time automated resolution of most of the RAL expressions defined in [1].** Only those assignments related to information coming from data objects of the process are not yet implemented, since the mechanism to access data is ad-hoc to the BPMS in which the process is run, and thus we did not consider this a fundamental feature to include in the prototype. Run-time implementation is delivered as a library for Activiti, a light-weight open-source WF and BPMS.
4. **Flexibility in the organizational model** against which RAL expressions are solved. The only requirement is that the model must comply with the organizational meta model used by RAL [1].
5. **Usability.** In order for module RACI2BPMN to be executed, the user must fill in the RACI matrix, and configure the binding information according to his/her needs. The rest of the procedure is automatic. As for RAL Solver, the use of both the plug-in for Oryx and the code for Activiti is straightforward.

[1] http://activiti.org/

For the former, a few instructions can be found at `http://www.isa.us.es/cristal/`. The latter is totally transparent to the user of the BPMS.

6. **Extensibility capabilities.** More analysis operations over RAL assignments can be easily added by composing operations already implemented by current DL reasoners (e.g. HermiT, or Pellet), as stated in [2].
7. **Re-usability.** The core of the RAL Solver tool (i.e. RAL Analyser) can be used in other platforms, since it provides a simple and well-defined interface.

However, CRISTAL has also some limitations. The efficiency of complex and/or compound resource assignments may not be good enough due to the inference operations the DL reasoner has to perform.

3.1 Significance to the BPM Field

CRISTAL offers innovative features with respect to the (human) resource management capabilities provided by current BPM notations and systems.

First, to the best of our knowledge, RACI2BPMN is the first tool focused on the automated introduction of RACI information in a process model, generating RACI-aware BP models that can be executed with no need of changes.

Second, regarding RAL, the RAL Solver tool provides BP modelling languages with a more expressive mechanism to assign resources to tasks. Furthermore, as far as we know, RAL is the first resource assignment language that offers automated analysis capabilities at design time that are built in the language itself. As for run time, it is a fact that most of the BPMSs existing at present have resource assignment mechanisms based basically on assigning users or groups of users (sometimes also roles) to the BP activities (e.g. Activiti, jBPM, Intalio—BPMS). Other tools such as YAWL are more expressive because they are supported by a more powerful organizational meta model. However, YAWL does not rely on a specific language for resource assignments, and uses ad-hoc mechanisms to assign resources and resolve the assignments instead. Therefore, we believe RAL outperforms the current scope, both at design time and at run time.

4 Availability

Further information about RAL, the descriptions of the tools, user instructions, and downloadable example files can be found at `http://www.isa.us.es/cristal`. The source code and executable files are available on demand.

5 Future Work

CRISTAL can be extended in different directions referring to resource management in BPs. Some future work consists of adding the proper functionality to detect at design time potential allocation problems that can arise at run time,

and which are derived from the control flow of the process, e.g. empty sets of potential performers due to the definition of a Binding of Duties (BoD) constraint between two activities that belong to different branches of an XOR gateway.

Another possible line to extend the tool is about dealing with data together with resources, e.g. to automatically generate Access Control Lists (ACLs) from a resource and data-aware BP model.

The optimization of RAL Analyser is also part of planned work.

Acknowledgements

We want to thank ISA group's development team for the support provided, especially Ana Belén Sánchez and Edelia García.

References

1. C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, "RAL: A High-Level User-Oriented Resource Assignment Language for Business Processes," in *Business Process Management Workshops (BPD'11)*, pp. 50–61, 2012.
2. C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, "Defining and Analysing Resource Assignments in Business Processes with RAL," in *ICSOC*, pp. 477–486, 2011.
3. M. Weske, *Business Process Management: Concepts, Languages, Architectures*. Springer, 2007.
4. "BPMN 2.0," recommendation, OMG, 2011.
5. M. Smith, "Role And Responsibility Charting (RACI)," in *Project Management Forum (PMForum)*, p. 5, 2005.
6. G. Decker, H. Overdick, and M. Weske, "Oryx - an open modeling platform for the BPM community," in *Business Process Management (BPM)*, pp. 382–385, 2008.
7. C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, "Automated Resource Assignment in BPMN Models using RACI Matrices," in *OTM Conferences (CoopIS'12)*, p. In press, 2012.

Author Index