8th International Conference on
Business Process Management

BPM2010

BPM 2010 Demonstration Track

8th International Conference on
Business Process Management BPM'10

14-16 September 2010, Hoboken, New Jersey
http://www. bpm2010.org

Online Proceedings

Marcello La Rosa
(editor)
m.larosa@qut.edu.au

Preface

The BPM 2010 Demonstration Track was held in conjunction with the 8th International Conference on Business Process Management (BPM 2010) on 14th-16th September 2010, in Hoboken, USA. This track showcased 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.

We received 26 submissions, of which 11 were intended as demo proposals only, and 15 also included a demo paper. In total, we accepted 20 demo proposals, of which 11 were demo papers. These proceedings contain the demo papers. These papers: (i) clearly state how the presented tools are innovative, (ii) describe their significance to the field of BPM, (iii) list their main features and specify their maturity level, and (iv), if applicable, include brief descriptions of case studies performed using these tools, and pointers indicating where readers can find more information about these case studies.

We would like to thank the authors for their submissions, our Reviewing Committee for their hard work and for submitting their reviews on time, and the organizers of the BPM 2010 conference for their support which made this demo track possible.

<div align="right">

Brisbane, September 2010
Marcello La Rosa

</div>

Organization

Demo Chair

Marcello La Rosa
Queensland University of Technology
GPO BOX 2434, Brisbane
QLD 4001, Australia
Tel: +61 7 3138 9482
Fax: +61 7 3138 9390
E-mail: m.larosa@qut.edu.au
URL: www.marcellolarosa.com

Reviewing Committee

Claudio Bartolini, HP Research Center, United States
Christoph Bussler, Saba Software, United States
Anis Charfi, SAP Research CEC Darmstadt, Germany
Gero Decker, Signavio, Germany
Massimiliano de Leoni, University of Roma - Sapienza, Italy
Remco Dijkman, Eindhoven University of Technology, Netherlands
Luciano García-Bañuelos, University of Tartu, Estonia
Christian Güenther, Fluxicon, Netherlands
Dimka Karastoyanova, Stuttgart University, Germany
Rania Khalaf, IBM T.J. Watson Research Center, United States
Marcello La Rosa, Queensland University of Technology, Australia
Niels Lohmann, University of Rostock, Germany
Antonio Rito Silva, INESC/IST Technical University of Lisbon, Portugal
Nick Russell, Carba-Tec Pty Ltd, Australia
Stefanie Rinderle-Ma, University of Ulm, Germany
Boudewijn van Dongen, Eindhoven University of Technology, Netherlands
Barbara Weber, Innsbruck University, Austria
Matthias Weidlich, Hasso-Plattner Institute, Germany
Moe Wynn, Queensland University of Technology, Australia

Summary

Signavio-Oryx Academic Initiative

Matthias Kunze, Mathias Weske

Hasso Plattner Institute, University of Potsdam, Germany
{matthias.kunze,mathias.weske}@hpi.uni-potsdam.de

1   Background

Since 2007 we have seen increasing interest in using the Oryx process modeling framework[1] by academics. Many universities already use the platform in teaching to model, analyze, and simulate processes. While many courses focus on BPMN, also Petri nets and EPCs are used, all of which are supported by the system. The *Signavio-Oryx Academic Initiative* (SOAI) is a joint project of academic and industrial partners that aims at providing a mature process modeling platform and teaching material for researchers and lecturers, free of charge[2].

The SOAI builds on the technical foundation of the Web-based process modeling tool Oryx, an open and extensible platform for business process management [1]. Following recent trends in application design and delivery, SOAI offers comprehensive features for process modeling and management in the fashion of Software as a Service, superseding those of Oryx. Both, application logic and model data, reside on the Web: The modeling environment runs in a standard Web browser; each model is identified via a URL and can be obtained in different representations, e.g. pdf, xml, svg, and png. Thus, models can be shared by reference, i.e., bookmarks. This contributes to collaboration, since several modelers work on the same resource, rather than on copies. For each model a complete version history is available.

Signavio is a startup company from members of the Business Process Technology group at Hasso Plattner Institute. The Signavio Process Editor is a commercial process modeling and analysis tool that is based on Oryx. The company hosts, supports, and maintains the technical platform of the SOAI.

The initiative further comprises a team of academic partners: the Stevens Institute of Technology in Hoboken (USA), the Technische Universiteit Eindhoven (Netherlands), the Queensland University of Technology in Brisbane (Australia), the Universität Stuttgart, the Humboldt-Universität zu Berlin, and the Hasso Plattner Institute, Universität Potsdam (Germany). Together, we strive to promote the platform and contribute to a high quality body of teaching material and tools in the BPM field.

---

[1] cf. http://oryx-project.org
[2] cf. http://www.signavio.com/academic

2 Use Cases

The Signavio-Oryx Academic Initiative has been established to support researchers and lecturers in their work, and thus, provides assistance in lecture assignment management and team-based collaboration.

2.1 Lecture Assignments

In university curricula, specific subjects, e.g., business process management, are held as self-contained courses or embedded in programs of broader scope, e.g., Requirements Engineering. Students attend lectures and refer to textbooks; however, practical experience and relevant exercises are essential to master a subject.

The SOAI platform provides a means to conduct practical exercises, illustrated in Figure 1, through a comprehensive set of resources and tools. A lecturer assembles an exercise sheet from a set of assignments according to their needs. The SOAI provides a collection of freely available exercises to use or inspire the lecturer in this task. Typically, the exercise sheet will be distributed electronically to students.
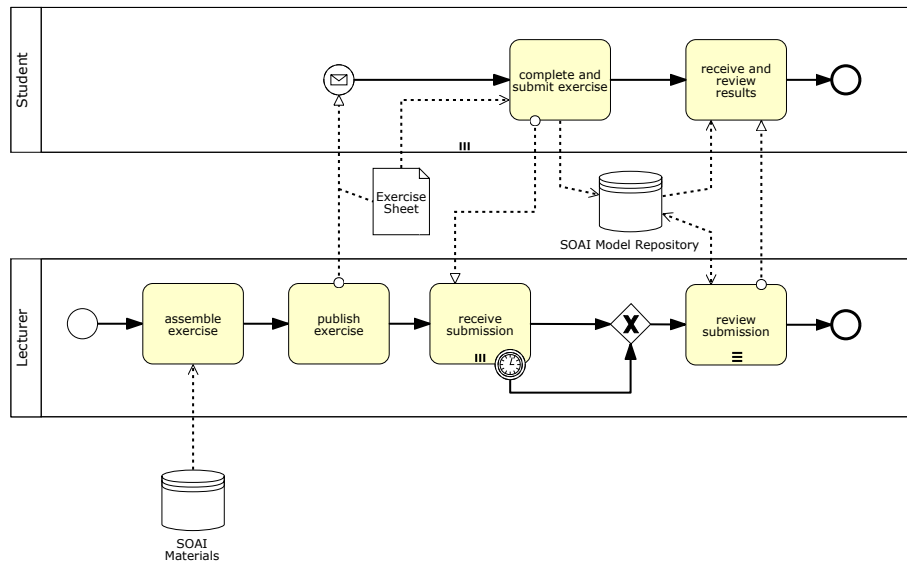

Fig. 1. Use Case: Lecture Assignment Management (BPMN 2.0)

Instead of downloading a tool, solving the assignments, exporting, printing, and handing in the results on paper, students solve the assignment in the Web-based process model editor of the SOAI—the model will be saved online and is

accessible via a Web browser. Upon completion of their assignments, the students submit their results by inviting the lecturer to review their results online, granting them access to read and comment on the diagrams. The lecturer gives feedback on the solution of the assignments by appending comments to models and model elements, pointing to exemplary solutions or flaws therein.

2.2 Team-based Collaboration

In addition to inviting people to review and comment on the diagram, the SOAI platform offers workspaces for collaboration. A workspace is a virtual directory that enables multiple people in different locations to collaborate on a common project and develop models together. A version history prevents lost updates and offers insight into the evolution of process models over time.
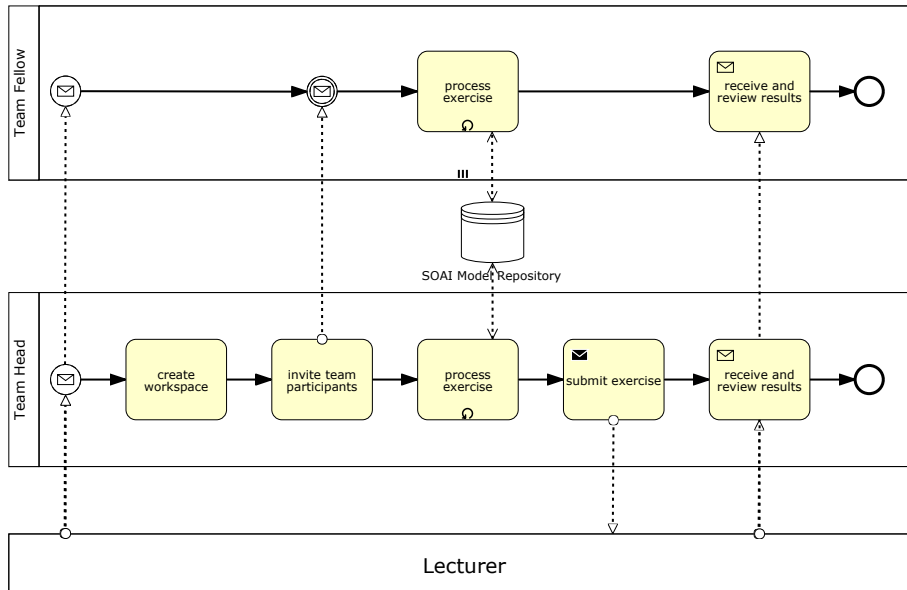
Fig. 2. Use Case: Team-based Collaboration (BPMN 2.0)

Figure 2 shows an extension of the previous use case: Rather than solving assignments individually, students form teams to submit a collaborative solution. Thus, the multiple instance pool *student* from Figure 1 has been split into a *team head* and several *team fellows*. Upon reception of the assignment sheet by each of them, the team head creates a workspace and invites his fellow students, thereby granting access rights to edit the models in the workspace. After jointly working on the diagram—manipulating it and reviewing it by means of the discussion

feature—the team head submits the results to the lecturer, by inviting them and sharing read access to the models. Again, all participants receive the review results from the lecturer in due course.

3 Platform Overview

In the last eight months, i.e., from the platform's launch up to the time of writing this paper, the Signavio-Oryx Academic Initiative has received broad interest among academics. More than 1000 persons from 250 universities are using the platform regularly, and over 3500 models have been created.

3.1 Architecture

The SOAI platform comprises, at a high level, two components: the lecture material collection and the modeling tool, which in turn consists of a model editor, a model publisher, and a repository explorer, cf. Figure 3.
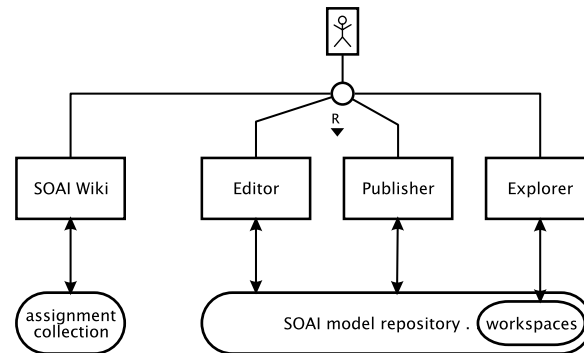
Fig. 3. SOAI Platform Architecture (FMC)

Assignment Collection. As already stated in the first use case, the initiative offers a body of exercise material with a particular focus on topics in business process management. The assignments are maintained in a wiki[3] that is accessible publicly and are provided for non-commercial use under the terms of the Creative Commons license[4]. The wiki contains further information and references towards the field of BPM and use of the platform.

---

[3] cf. `http://bpt.hpi.uni-potsdam.de/SignavioOryxAcademicInitiative/`
[4] i.e. *Creative Commons Attribution-Noncommercial-Share Alike 3.0 License*

Editor. The model editor of the SOAI is, in its basics, the Oryx Editor [1], which has been adapted to fit the initiative's setting and to satisfy high quality requirements. Work from many contributors has made its way into the platform and is offered to the users, as will future developments, especially with regard to new scientific features and modeling languages. By offering the platform to a broad audience, we also hope to exchange new inspiring ideas and encourage academic and scientific institutions to participate and contribute their work to the platform.

Publisher and Explorer. Signavio provides the technological basis for the initiative to run, including the hardware infrastructure, provides continuous maintenance of the platform, and further contributed additional features for collaboration and model management to the initiative.

The model explorer provides general functionality to load and store models in the repository, as well as a rich set of tools to report, export, and augment stored models. It is also the basis for managing workspaces and granting access rights to view, comment, and edit diagrams of other users. The publisher offers a perspective to browse through revisions of a process model and attach comments to certain model elements or the model as a whole. This feature is designed to enable discussion at a fine grain right on the model, rather than in separate forums.

4 Conclusion and Outlook

The Signavio-Oryx Academic Initiative is an alliance of a number of BPM research groups and industrial partners, offering a comprehensive platform for teaching, including tools to conduct collaborative assignment processes, as well as a collection of exercise material and related information.

We envision that the usage of the platform yields a large process model collection that can be used for empirical research in modeling sciences in general, and business process management in particular. The samples promise to include a wide variety of expertise levels, and we envision a comprehensive collection of process variants, e.g., through relating models to assignments.

References

1. Gero Decker, Hagen Overdick, and Mathias Weske. Oryx - An Open Modeling Platform for the BPM Community. In Marlon Dumas, Manfred Reichert, and Ming-Chien Shan, editors, *BPM*, volume 5240 of *Lecture Notes in Computer Science*, pages 382–385. Springer, 2008.

$\mathcal{PN}av$: Process Navigator for the Design of New Business Process Models

Maya Lincoln and Avigdor Gal

Technion - Israel Institute of Technology
mayal@technion.ac.il, avigal@ie.technion.ac.il

Abstract. In this demonstration we introduce a prototype of $\mathcal{PN}av$, a process navigator that assists designers in designing new process models. To do that, $\mathcal{PN}av$ generates activity suggestions for the newly generated process models. The business logic for such suggestions is extracted from process repositories through the analysis of existing business process model activities.

1 Introduction

Enterprise process repositories contain hundreds of business processes, developed over the years to support enterprise activities. Such repositories contain a large number of activities that can be re-used when redesigning existing processes or whenever the need for new processes arises. Process modeling is considered a manual, labor intensive task, whose outcome depends on personal domain expertise with errors or inconsistencies that may lead to bad process performance and high process costs [3]. Hence, reuse of activities can save design time and support non-expert designers in creating new business process models.

In this demonstration we introduce a prototype of $\mathcal{PN}av$, a process navigator that assists designers in designing new process models. To do that, $\mathcal{PN}av$ generates activity suggestions for the newly generated process models. The business logic for such suggestions is extracted from process repositories through the analysis of existing business process model activities. Each activity is encoded automatically as a *descriptor*, using the PDC notation [2,1]. The collection of all descriptors formulates a descriptor space, and distances between every two space coordinates are calculated in terms of business process conduct proximity.

In the Process Descriptor Catalog model ("PDC") [2,1] each activity is composed of one action, one object that the action acts upon, and possibly one or more action qualifiers and object qualifiers. For example, the activity name"Manually complete a supplier maintenance form" is decomposed into (action='complete' , object='form' , action qualifier='manually' , object qualifier='supplier maintenance').

The model has two basic elements, namely objects and actions, and we delineate four taxonomies from them: an *action hierarchy model*, an *object hierarchy model*, an *action sequence model* and an *object lifecycle model*. The business action and object taxonomy models organize a set of activity descriptors according

to relationships among business actions and objects both hierarchically and in terms of execution order.

An example from the Oracle Business Model (OBM)[1] is given in Fig. 1 and Fig. 2, showing a section of the business action and object taxonomy models as a set of activity descriptors according to the relationships among business actions and objects both longitudinally (hierarchically) and latitudinally (in terms of execution order), as detailed next.
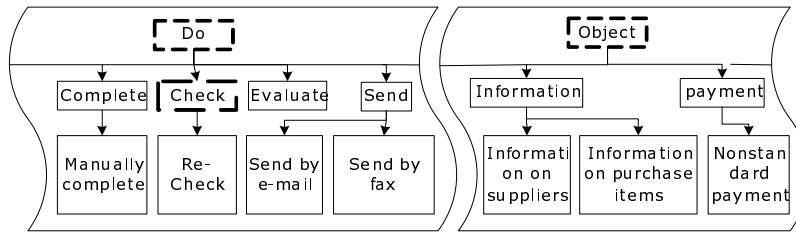
Fig. 1. Segments of the action hierarchy model and the object hierarchy model extracted from the OBM for procurement processes.
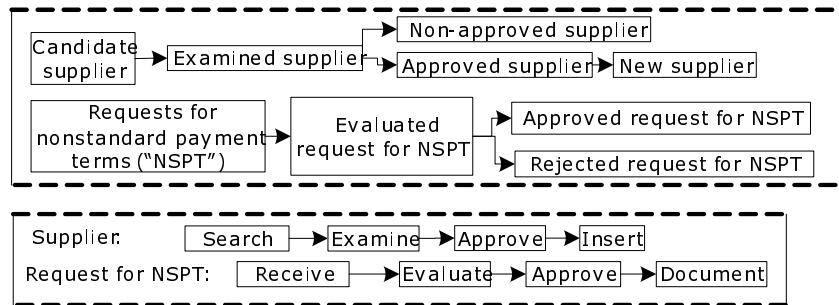
Fig. 2. Segments of the action sequence model and the object lifecycle model extracted from the OBM for procurement processes.

The longitudinal dimension of actions and objects is determined by their qualifiers. To illustrate the longitudinal dimension of the OBM workflows, a segment of the action hierarchy model and a segment of the object hierarchy model, both related to procurement processes, are presented in Fig. 1. To illustrate the latitudinal dimension of the OBM process repository, a segment of the action sequence model and a segment of the object lifecycle model are presented in Fig. 2.

---

[1] http://www.oracle.com/applications/tutor/index.html.

Based on the activity decomposition model, it is possible to visualize the operational range of a business process model as a descriptor space comprised of objects and actions, related to each other and among each other in different relationship types. A navigation within this space can be a powerful tool for analyzing and utilizing the underlying business process knowledge encapsulated within a business process repository. More details of the descriptor space and how to navigate it can be found in [1].

The demo is intended for both academics, interested in new techniques for designing business process models and practitioners, interested in state-of-the-art technology for design support tools. The tool can help automate the reuse of constructs gathered from predefined process models. Such a tool saves design time and supports non-expert designers in creating new business process models. The proposed software tool, can be used in real-life scenarios, yet several research and development are required in order to enable a more commercialized version of the tool.

2 Maturity

$\mathcal{PN}av$ implements a client-server architecture, in which the client is responsible for presenting and collecting data from the user and the server is responsible for processing the user's input data and suggesting directions for advancing the design process. Server side logic is implemented in PHP using a MySql database. The client runs within an Internet browser and is implemented in HTML and JavaScript, with AJAX calls to the server.
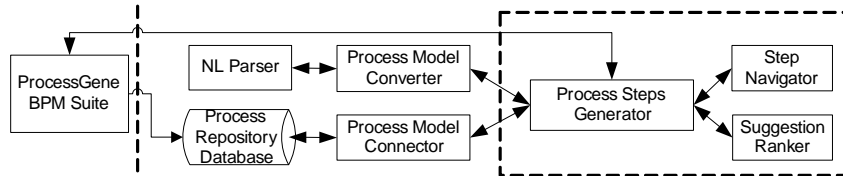
Fig. 3. $\mathcal{PN}av$ high-level architecture.

The server side high-level architecture includes five main components (see Fig. 3): (a) the navigator, responsible for managing and orchestrating the process design mechanism; (b) the process repository database that contains the existing business process repository, in terms of activity descriptors and object and action taxonomies; (c) the process model connector, which provides an interface for communicating with the process repository database; (d) the process model converter, responsible for converting inputted business process repositories into a normalized data structure as saved in the process repository database. Currently, our system supports the conversion of repositories expressed in BPEL

or BPMN; (e) the Natural Language (NL) parser, an existing web service for decomposing sentences into linguistic components. The parser we use is called "the Stanford Parser[2]. This web service decomposes activity names into descriptor components.

The navigator is further decomposed into four main components: (a) the process steps generator, responsible for producing suggested activities for each design phase. It communicates with all other components and presents the user at each stage with options for advancing the design process; (b) the step navigator, which is responsible for navigating in the process model database, and for retrieving a list of relevant activity options; (c) the suggestion ranker, responsible for ranking the suggested activity options at each stage.

The navigator is designed to connect to the ProcessGene BPM suite and to assist designers in designing new process models. Each time a user opens a new process model in ProcessGene[3], and defines the new process model name, $\mathcal{PNav}$ is activated and suggests the user to use its services. Once the user decides to use $\mathcal{PNav}$, she is guided in a step-by-step procedure that advises and supports the creation of the new process model.

We have conducted experiment sets with $\mathcal{PNav}$ using two case studies[1]. The first experiment set was based on an aviation process repository that covers airport activities starting from the passenger's entry to an airport, through document handling and security checks and terminating as the passenger boards the airplane[4]. The second experiment set was based on the Oracle Business Model, which serves for our demonstration script and will be discussed in details next.

$\mathcal{PNav}$ is currently installed and works well on a workstation running Windows XP, IIS6, PHP 4.8 and MySQL 5.0. This workstation serves as the server. A client, running Internet Explorer as the application container and presentation layer, will be available at the Demo site.

3 Script

We demonstrate the applicability of our ideas using 14 processes from the Oracle Business Model. Nine business processes are taken from the "Procurement" category, containing altogether 96 activities and five business processes are taken from the "Inventory" category, containing altogether 31 activities. The "Procurement" data set contains related, sequential activities and therefore encapsulates a focused operational area. The "Inventory" data set encapsulating a loosely coupled business logic regarding an extended business area.

Using the selected 14 processes we created a process repository database. The demo user shall provide a starting activity and then interact with $\mathcal{PNav}$ to receive a ranked list of suggested activities, refine the suggestions and move

---

[2] http://nlp.stanford.edu:8080/parser/index.jsp

[3] http://www.processgene.com

[4] Many thanks to Samia Mazhar and the BPM Group at QUT for providing access to the aviation process data.

on to the next activity. It should be noted that the tool offers only activities –
designer needs to add junctions.

3.1 Example for using $\mathcal{PN}av$ for Designing a New Process Model

To illustrate the proposed software tool we present a short example from the
field of procurement. The newly designed process, "Verify supplier details", is
related to the procurement field, but is not covered by the OBM. Its goal is to
verify whether the supplier details, as declared in the new supplier form, are
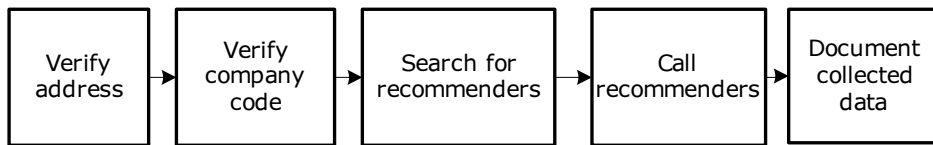correct.

Fig. 4. The new designed process diagram for "Verify supplier details".

The example supports the design of a new business process for: "Verify sup-
plier details." The generated output (new process model) of this example is
illustrated in Fig. 4 as a YAWL diagram. The design process starts when the
(human) process designer inserts into $\mathcal{PN}av$ the name of the new process ("Verify
supplier details") which is then translated automatically into the following pro-
cess descriptor: (action="verify", action qualifier=null, object="details", object
qualifier="supplier") (see Fig. 5a) and determines that the first activity is: "Ver-
ify address." Respectively, the process delineator searches the descriptor space,
looking for next activity possibilities. The result set includes the following ac-
tivities: "[1] Check supplier," "[2] Verify supplier" and "[3] Verify address" (see
Fig. 5b). The designer selects the option "Verify address" and decides that this
activity is suitable.

The design process continues with four more design phases. The second phase
required a refinement for the option "Search for additional data" - which was
suggested as the next activity after "Verify company code." The refined option
list includes the option: "[1] Search for recommenders," and this option was
selected by the designer. Note that this activity was not represented "as is" in
the business process repository.

The designer now wishes to design the new business process: "Review invoices
to prevent fraud." An interesting observation in this design process is that the
designer selects more often next step activities that share the same action applied
on sibling objects. For example, the activity "Check signature" was followed
by "Check date" and "Check payment terms." The business logic behind this
phenomenon is that this process expresses a more independent business conduct
in which there is only one party (the reviewer) which operates on one item (the
invoice).

(a) The designer's request for designing the new process : "Verify supplier details"

(b) The third phase in designing the new process: "Verify supplier details"
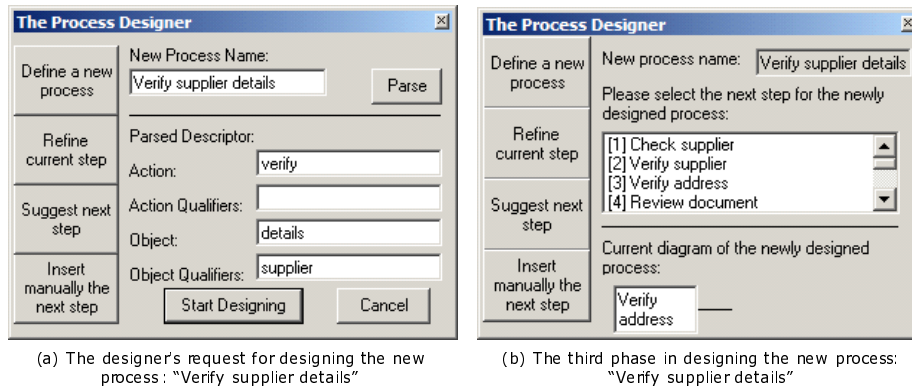
Fig. 5. The designer's request for designing the new process: "Verify supplier details".

References

1. M. Lincoln, M. Golani, and A. Gal. Machine-assisted design of business process models using descriptor space analysis. In *Proceedings of the eigth International Conference on Buisness Process Modeling (BPM'2010)*, Hoboken, NJ, USA, September 2010.
2. M. Lincoln, R. Karni, and A. Wasser. A Framework for Ontological Standardization of Business Process Content. *International Conference on Enterprise Information Systems*, pages 257–263, 2007.
3. D. Muller, M. Reichert, and J. Herbst. Data-driven modeling and coordination of large process structures. *Lecture Notes in Computer Science*, 4803:131, 2007.

Enabling Process Support for Advanced Applications with the AristaFlow BPM Suite

Andreas Lanz[1], Ulrich Kreher[2], Manfred Reichert[1], and Peter Dadam[1]

[1] Institute of Databases and Information Systems, University of Ulm, Germany
[2] AristaFlow GmbH, Ulm, Germany

Abstract. A process-aware information system (PAIS) will be not accepted by end users if its software clients do not support their native workflows or are too complex for them. When implementing business processes based on process management technology important issues are, therefore, how end-users can participate in the execution of the processes and how this can be accomplished as intuitively as possible. This becomes extremely important if high flexibility demands need to be fulfilled during process execution, while PAIS robustness and error safety need to be assured. In this software demonstration we show how the AristaFlow BPM Suite – an adaptive process management system developed by us – was applied to challenging applications in domains like healthcare, logistics, disaster management, and software development. The implementation of adaptive software clients in these different applications particularly proves the benefits provided by an open application programming interface (API) as offered by AristaFlow.

1 Introduction

In many domains IT support can benefit from BPM technologies and the PAIS based on them. However, in many cases still specialized application systems are developed for accomplishing tasks being similar to process management functions. One reason for this is that available PAIS often cannot be adapted to domain-specific problems. This leads to inflexible systems with hard-wired processes where small changes in the underlying process often require changes to the whole system. Especially in domains with high flexibility demands, existing BPM technology is not used due to its brittleness and inflexibility. Another side effect of these specialized systems is that processes are only known implicitly and are not modeled explicitly, making it hard to analyze and improve them.

During the last decade we developed the ADEPT2 process management technology [1–3]. Due to the high interest companies had in ADEPT2 we transferred it into an industrial-strength process management system called *Arista-Flow BPM Suite* [4–6]. One of our basic goals is to enable robust and flexible PAIS for a large number of processes from different domains. Recently, together with partners, we applied the AristaFlow BPM Suite to a variety of challenging application domains like healthcare, disaster management, logistics, and software

engineering. In our software demo we will show some of the tools and software clients developed in this context using the AristaFlow Open API.

Section 2 presents results from domain-specific projects in which we applied the AristaFlow BPM Suite. Section 3 describes the AristaFlow Open API. We conclude with a short summary and outlook in Section 4.

2 Applying AristaFlow to Advanced Applications

Generally, domain experts only have little or no IT knowledge. Therefore, the standard clients as offered by existing BPM systems are inappropriate for them and domain-specific tools and PAIS interfaces have to be developed. This means that working habits of end-users need to be analyzed and specific tools be developed. These tools must provide exactly that functionality as required by the respective domain expert and it must present this functionality in a way the user can easily handle, i.e., the respective systems needs to be tailored to the users' needs and the given application domain.

Recently, AristaFlow BPM Suite was applied to a number of challenging domains. In each project sophisticated BPM tools were implemented supporting end-users in their work as best as possible.

2.1 Application 1: Emergency Management

Domain. The project for *process-aware, cooperative emergency management of water infrastructures* [7] aimed at improving and supporting emergency management for flood responses through new IT methods. During the project, procedures and courses of actions were analyzed, and results were mapped to formal process models. On the basis of an organizational model, the activities of the process models were assigned to the responsible parties, thus enabling the involved organizations to act faster and in a more coordinated way.
Applying AristaFlow. AristaFlow was used to manage and control the procedures and tasks during flood events as well as the corresponding information flow. Thus, it supported the responders in planning and executing flood response operations. Fig. 1 shows a domain-specific PAIS user interface that was implemented using the AristaFlow Open API. In particular, users may order resources and deploy them to emergency locations.
Discussion. AristaFlow supports responders in accomplishing their operations in a coordinated, but flexible way. One important aspect was to provide the necessary flexibility, while ensuring robustness and error safety of the PAIS.

2.2 Application 2: Healthcare and Logistics

Domain. Healthcare and logistics are both characterized by high flexibility demands. Additionally, both require tools that are easy to use since domain specialists have no IT knowledge. By supporting domain-specific views on processes
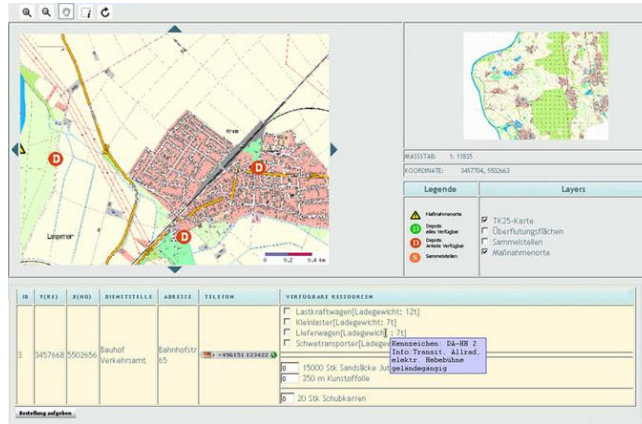
Fig. 1. User interface for interacting with disaster management processes [7]

(e.g., clinical pathways) and services, the *SPOT* project [8] (*Service-based technologies for orchestrating PrOcesses in logisTics and healthcare*) enables end-users to actively shape the different phases of the process as well as the service life cycle. Another important aspect concerns application integration. In both domains system integrators are confronted with heterogeneous, autonomous applications, to be integrated in a process-oriented way.

Applying AristaFlow. The SPOT project team selected AristaFlow BPM Suite as implementation platform for several reasons: Besides its correctness-by-construction principle, its strict modular design and its service-oriented architecture were basic points in favor of AristaFlow. This enabled the service-oriented provision of advanced process support features and simplified the integration of existing application systems. Fig. 2 shows a mobile client for patient assistance in a hospital being connected to an AristaFlow process server. For representing clinical pathways the SPOT project uses editable tree structures, which are automatically mapped to (block-structured) AristaFlow process models. In particular, adaptations within such tree structure are translated into corresponding changes of the underlying process model and can be automatically applied to the considered instance using ad-hoc changes. This empowered domain experts (e.g. nurses) to change processes at a high level of abstraction.

Discussion. Again, the AristaFlow Open API facilitated the integration of our process engine into the overall architecture of the SPOT system. Additionally, features like robust process execution and user assistance in connection with ad-hoc process instance changes where considered being extremely useful features.

2.3 Application 3: Software Engineering

Domain. Due to the dynamic nature and high degree of collaboration, communication, and coordination inherent in software engineering projects, automated workflows can assist overburdened software engineers by providing orientation

and guidance. Yet, since there are so many different kinds of issues with ambiguous and subjective delineation, it is difficult and burdensome to universally and correctly model them in advance. This will also lead to workflows of considerable size and complexity. The Q-Advice project [9] tries to alleviate this by starting with a basic and simple workflow for each case and then, utilizing context information and ArisaFlow BPM Suite, dynamically extends it with activities matching the current situation.

Applying AristaFlow. The Q-Advice project uses the AristaFlow BPM Suite as its process module. Thereby it makes heavy use of the AristaFlow API for automatically constructing and adapting process models as well as for developing specialized activity components. Based on context information an issue workflow is automatically, dynamically, and uniquely constructed for every software engineering issue. The activities of the workflow are then distributed to the responsible users based on the organization model maintained by AristaFlow. Fig. 3 shows part of the Q-Advice user interface where in the lower section the user can see his current task as well as the next upcoming tasks. In the upper section additional information is provided by the framework.

Fig. 2. SPOT project: Mobile client for personal treatment plan [8]
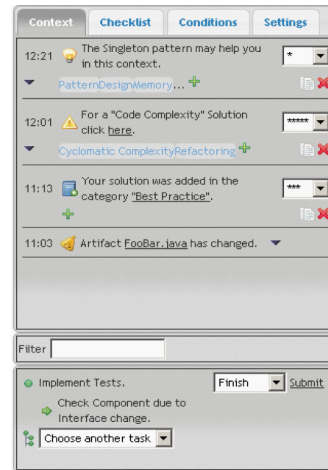
Fig. 3. Q-Advice user interface [9]

Discussion. Q-Advice makes it possible to provide situational and tailored support and guidance for software engineers. In particular the workflows resulting from the Q-Advice approach are much simpler than pre-modeled workflows would be. Thereby Q-Advice makes use of the advanced change facilities of AristaFlow and integrates them into its framework. By using AristaFlow BPM Suite it became possible to hide the inherent complexity of process-orientation, dynamic process changes, and flexible task management from users; i.e., all complex things are performed "beneath the surface" within the AristaFlow system.

2.4 Other Projects using the AristaFlow BPM Suite

Over the last years, more than 20 other groups from academia and industry applied the ADEPT/AristaFlow process management technology in the context of research projects. In the E-Commerce domain, for example, CONSENSUS [10] offered a flexible support system for e-negotiations based on parameters like quality, delivery, warranty, and financial terms. In the MTCT project [11] a process-aware system architecture for the processing of client requests (CRs) in container transportation was realized. AgentWork and HematoWork [12] offered a rule-based system for automatically adapting clinical process instances in case of exceptional events. These are just some of the projects that used the AristaFlow BPM Suite (or its academic counterpart ADEPT) to realize domain-specific PAIS.

3 The AristaFlow Open API

Due to its Open API as well as its strict modular and service-oriented design (see Fig. 4), AristaFlow can be easily applied and adapted to different application domains. This way we enable integration of advanced process support functions into domain-specific PAIS as well as provision of domain-specific client, service and activity implementations.

All services of the upper layers of the overall architecture of the AristaFlow BPM Suite depicted in Figure 4 can be utilized by domain-specific client applications and thus be easily integrated using the AristaFlow Open API. A functional overview of the different services is provided in [13].

The development of the platform itself was driven by the development of its API. Even the built-in client applications of AristaFlow (including, e.g., the process editor) are just implementations of this Open API. The structure of the API also enables easy integration of different execution platforms and client types (e.g., rich-clients, web-clients, or mobile devices) and further enables integration into specialized applications. Last but not least, ad-hoc changes cannot only be applied by system administrators, but also by end-users. Thereby one or more low-level change operations can be combined to form higher-level change patterns in order to perform domain-specific operations.
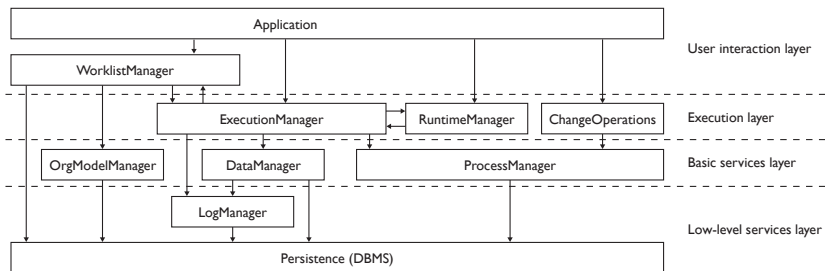

Fig. 4. Basic architecture of the AristaFlow BPM Suite

4 Summary and Outlook

Domain-specific solutions can be easily developed based on the AristaFlow Open API. This API is not limited to solely high-level client applications, but includes all levels of the architecture; e.g. new services can be integrated easily and even the process and application meta models themselves can be extended (see [13] for more information on this topic).

References

1. Reichert, M., Rinderle-Ma, S., Dadam, P.: Flexibility in process-aware information systems. In: Transactions on Petri Nets and Other Models of Concurrency (ToPNoC). Volume 5460 of LNCS. Springer (2009) 115–135
2. Reichert, M., Rinderle, S., Dadam, P.: ADEPT workflow management system: Flexible support for enterprise-wide business processes. In: Proc. 1st Int'l Conf. on Business Process Management (BPM '03). Volume 2678 of LNCS. (2003) 371–379
3. Reichert, M., Rinderle, S., Kreher, U., Dadam, P.: Adaptive process management with ADEPT2. In: Proc. Int'l Conf. on Data Engineering (ICDE'05), IEEE Computer Society Press (2005) 1113–1114
4. Dadam, P., Reichert, M.: The ADEPT project: A decade of research and development for robust and flexible process support - challenges and achievements. Computer Science - Research and Development **22** (2009) 81–97
5. Lanz, A., Reichert, M., Dadam, P.: Making business process implementations flexible and robust: Error handling in the AristaFlow BPM Suite. In: Proc. CAiSE'10 Forum. (2010)
6. Reichert, M., Dadam, P., Rinderle-Ma, S., Lanz, A., Pryss, R., Predeschly, M., Kolb, J., Ly, L.T., Jurisch, M., Kreher, U., Göser, K.: Enabling Poka-Yoke workflows with the AristaFlow BPM Suite. In: CEUR proceedings of the BPM'09 Demonstration Track, Business Process Management Conference 2009 (BPM'09). (2009)
7. Wagenknecht, A., Rüppel, U.: Improving resource management in flood response with process models and web GIS. In: 16th TIEMS 2009. (2009) 141–151
8. Frauenhofer ISST: SPOT Project. http://www.spot.fraunhofer.de/ (accessed 13.07.2010) (2010)
9. Oberhauser, R.: Towards automated test practice detection and governance. In: Proc. Conference on Advances in System Testing and Validation Lifecycle. (2009) 19–24
10. Bassil, S., Benyoucef, M., Keller, R., Kropf, P.: Addressing dynamism in e-negotiations by workflow managament systems. In: Proc. Conference on Database and Expert Systems Applications (DEXA'02), IEEE Computer Society (2002) 655
11. Bassil, S., Keller, R., Kropf, P.: A workflow-oriented system architecture for the management of container transportation. In: Proc. Business Process Management Conference (BPM'04). Volume 3080 of LNCS. (2004) 116–131
12. Müller, R., Rahm, E.: Dealing with logical failures for collaborating workflows. In: Proc. 4th Int'l Conf. Cooperative Information Systems. Volume 1901 of LNCS. (2000) 210–223
13. Reichert, M., Dadam, P., Rinderle-Ma, S., Jurisch, M., Kreher, U., Goeser, K.: Architecural principles and components of adaptive process management technology. In: Process Innovation for Enterprise Software. Koellen-Verlag (2009) 81–97

MarcoFlow: Modeling, Deploying, and Running Distributed User Interface Orchestrations

Florian Daniel, Stefano Soi, Stefano Tranquillini, Fabio Casati

University of Trento, Povo (TN), Italy
{daniel,soi,tranquillini,casati}@disi.unitn.it

Chang Heng, Li Yan

Huawei Technologies, Shenzhen, P.R. China
{changheng,liyanmr}@huawei.com

Abstract. This paper introduces the idea of *distributed orchestration of user interfaces* (UIs), an application development approach that allows us to easily bring together UIs, web services, and people in a single orchestration logic, language, and tool. The tool is called *MarcoFlow*, and it covers three main phases of the software development lifecycle: design (by means of a dedicated, visual editor), deployment (by means of a set of code generators), and execution (by means of a distributed runtime environment for UI orchestrations). MarcoFlow targets the development of *mashup-like applications that require (distributed) process support* and, hence, targets researchers and practitioners interested in mashups, lightweight process design, web services, and innovative (and free) ways of providing process support.

1 Introduction

After workflow management (which supports the automation of business processes and human tasks) and service orchestration (which focuses on web services at the application layer), web mashups [1] feature a significant innovation: **integration at the UI level**. Besides web services or data feeds, mashups indeed reuse pieces of UIs (e.g., content extracted from web pages or JavaScript UI widgets) and integrate them into new web pages or applications. While mashups therefore manifest the need for reuse in UI development and for suitable UI component technologies, so far they produced rather simple applications consisting of one web page and of little utility.

This demo complements the concepts and solutions introduced in [2], where we argue that there is a huge spectrum of applications that demand for development approaches that are similar to those of mashups but that go far beyond single page applications and in fact support multiple pages, multiple actors, complex navigation structures, and – more importantly – process-based application logic or navigation flows. We call this type of applications **distributed UI orchestrations**, as (i) both components and the application itself may be distributed over the Web, (ii) in addition to traditional web services we also integrate novel JavaScript UI components, and (iii) services and UIs are orchestrated in an integrated fashion.

Challenges and contributions. Developing distributed UI orchestrations implies the coordination of individual actors and the development of a *distributed* user interface *and* service orchestration logic. Doing so requires (i) understanding how to *componentize UIs and compose them into web applications*, (ii) defining a logic that is able to *orchestrate both UIs and web services*, (iii) providing a language and tool for *specifying distributed UI compositions*, and (iv) developing a runtime environment that is able to *execute distributed UI and service compositions*.

Innovativeness of the tool. As of today, there is no single development instrument that answers these challenges and allows one to develop UI orchestrations using one language and one environment only. *BPEL* [3] focuses on web services only. *BPEL4People* [4] adds human tasks and actors as first-class concepts, but without supporting the development of suitable UIs. *Model-driven web design instruments*, such as WebRatio [5] or VisualWade [6], allow the development of advanced web applications, without however facilitating reuse of UI components sourced from the Web. *Portals and portlets* [7], instead, focus specifically on reuse, but they fail in supporting service integration and process flows. *Mashup tools* [1] support the integration of UIs and of services, but they typically do not support complex orchestration patterns (if not hand-coded). In mashArt [8], we did some first steps into that direction, but without considering multi-user and distributed UI support.

Significance to the BPM field. With MarcoFlow, we go one step beyond state-of-the-art BPM and service composition and propose an original model, language and running system for the composition of distributed UIs. The approach brings together UIs, web services and people in a single orchestration logic and tool and supports the development of mashup-like applications that require (distributed) process support, a kind of application that so far was not supported by BPM practices and software.

2   Distributed UI Orchestration

The **key idea** to approach the coordination of (i) UI components inside web pages, (ii) web services providing data or application logic, and (iii) individual pages (as well as the people interacting with them) is to split the coordination problem into two layers: *intra-page UI synchronization* and *distributed UI synchronization and web service orchestration*. UIs are typically event-based (e.g., user clicks or key strokes), while service invocations are coordinated via control flows. In this demo and in [2], we show how to describe UI components in terms of standard WSDL descriptors, how to bind them to JavaScript, and how to extend the standard BPEL language in order to support the two above composition layers. We call this extended language *BPEL4UI*.

Figure 1 shows the simplified meta-model of **BPEL4UI**. Specifically, the figure details all the new modeling constructs necessary to specify UI orchestrations (gray-shaded) and omits details of the standard BPEL language, which are reused as is by BPEL4UI. In terms of standard BPEL [3], a UI orchestration is a *process* that is composed of a set of associated *activities* (e.g., sequence, flow, if, assign, validate, or similar), *variables* (to store intermediate processing results), *message exchanges*, *correlation sets* (to correlate messages in conversations), and *fault handlers*. The services or UI components integrated by a process are declared by means of so-called

partner links, while *partner link types* define the roles played by each of the services or UI components in the conversation and the *port types* specifying the operations and messages supported by each service or component.
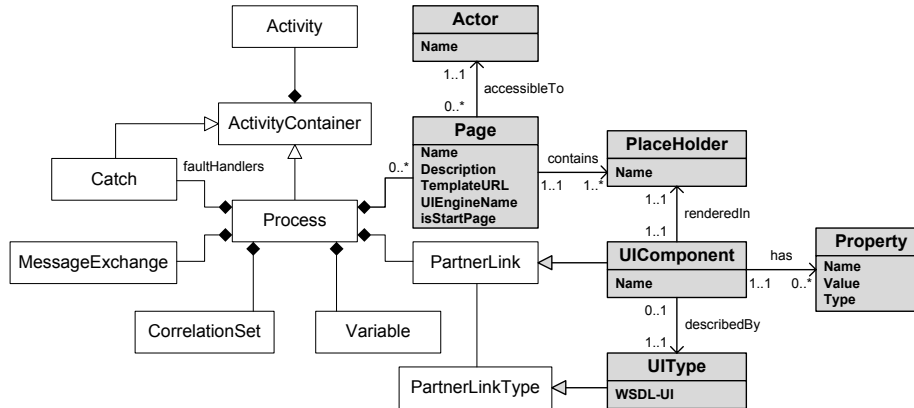

Figure 1 Simplified BPEL4UI meta-model in UML. White classes correspond to standard BPEL constructs; gray classes correspond to constructs for UI and user management.

Modeling UI-specific aspects requires instead introducing a set of **new constructs** that are not yet supported by BPEL. The constructs, illustrated in Figure 1, are: *UI type* (the partner link type for UI components), *page* (the web pages over which we distribute the UI of the application), *place holder* (the name of the place holders in which we can render UI components), *UI component* (the partner link for UI components), *property* (the constructor parameters of UI components), and *actor* (the human actors we associate with web pages).

It is important to note that although syntactically there is no difference between web services and UI components (the new JavaScript binding introduced into WSDL to map abstract operations to concrete JavaScript functions comes into play only at runtime), it is important to distinguish between services and UI components as their *semantics* and, hence, their usage in the model will be different. A detailed description of the new constructs and their usage can be found in [2].

As for the **layout** of distributed UI orchestrations, defining web pages and associating UI partner links with place holders requires implementing suitable HTML templates that are able to host the UI components of the orchestration at runtime. For the design of layout templates we rely on standard web design instruments. The only requirement the templates must satisfy is that they provide place holders in form of HTML DIV elements that can be indexed via standard HTML identifiers following a predefined naming convention, i.e., *<div id="marcoflow-left">... </div>*.

The main **methodological goals** in implementing our UI orchestration approach were (i) relying as much as possible on existing *standards*, (ii) providing the developer with only *few and simple new concepts*, and (iii) implementing a runtime architecture that associates each concern to the *right level of abstraction and software tool* (e.g., UI synchronization is handled in the browser, while service orchestration is delegated to the BPEL engine).
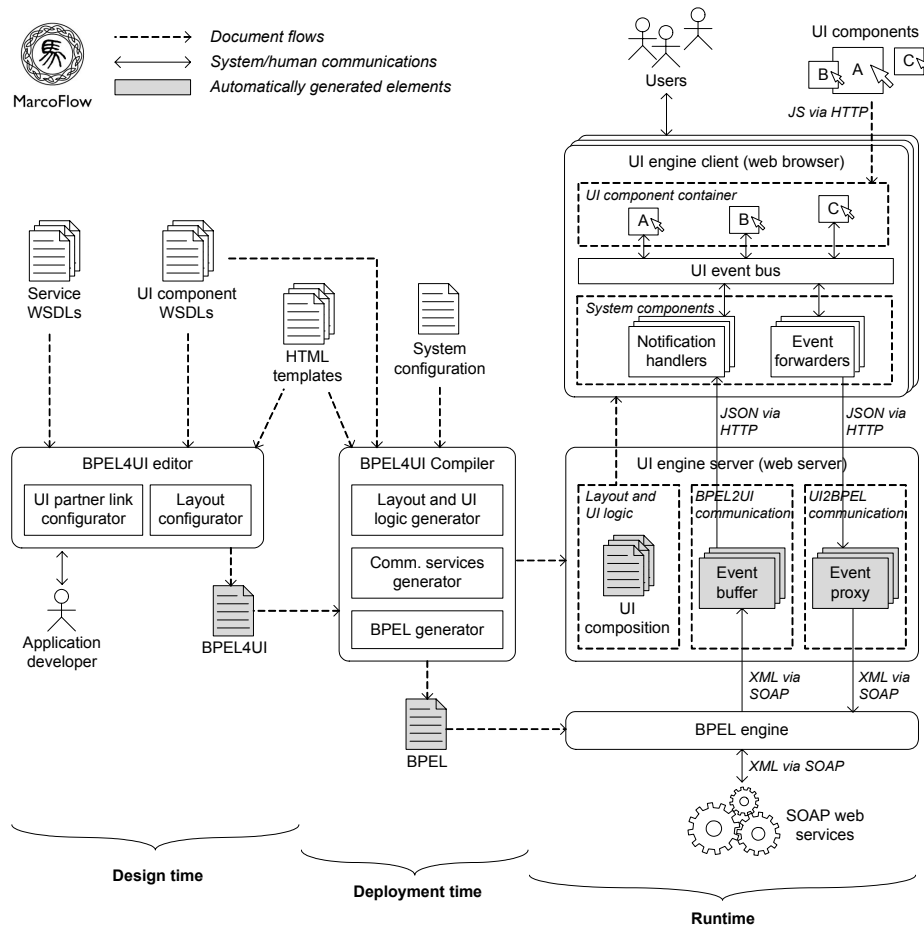
Figure 2 From design time to runtime: overall system architecture of MarcoFlow

3 The MarcoFlow Environment

Figure 2 shows the (simplified) architecture of the MarcoFlow environment, which aids the development and execution of distributed UI orchestrations. The architecture is partitioned into design time, deployment time, and runtime components, according to the three phases of the software development lifecycle supported by MarcoFlow.

The **design** part comprises the *BPEL4UI editor* that supports BPEL4UI [2], the composition language we use to specify distributed UI orchestrations. The editor is an extended Eclipse BPEL editor with (i) a panel for the specification of the pages in which UI components can be rendered and (ii) a property panel that allows the developer to configure the web pages, to set the properties of UI partner links, and to associate them to place holders in the layout.

The **deployment** of a UI orchestration requires translating the BPEL4UI specification into executable components: (i) a set of *communication channels* that mediate

between the UI components in the client browser and the BPEL engine; (ii) a *standard BPEL specification* containing the distributed UI synchronization and web service orchestration logic; and (iii) a set of *UI compositions* (one for each page of the application) containing the intra-page UI synchronizations. This task is achieved by the *BPEL4UI compiler*, which also manages the deployment of the generated artifacts in the respective runtime environments.

The **execution** of a UI orchestration requires the setup and coordination of three independent runtime environments: (i) the interaction with users and intra-page UI synchronization is managed in the client browser by an *event-based JavaScript runtime framework*; (ii) a so-called *UI engine server* runs the web services implementing the communication channels; and (iii) a *standard BPEL engine* manages the distributed UI synchronization and web service orchestration.

The MarcoFlow system shown in Figure 2 is fully implemented and running. A patent application for parts of the system has been filed.

4 Demo scenario

An example of how MarcoFlow can be used for the development of a distributed UI orchestration is available at *http://mashart.org/marcoflow/demo.htm*. The demo in form of a video illustrates in few minutes the main features of MarcoFlow in the context of a simple home assistance management application. Particular emphasis is given to the three development phases supported by the tool (design, deployment, and runtime) and to the use of the final application by the different actors involved in the distributed process logic.

References

1. J. Yu, B. Benatallah, F. Casati, F. Daniel. Understanding Mashup Development and its Differences with Traditional Integration. *IEEE Internet Computing*, Vol. 12, No. 5, September-October 2008, pp. 44-52.
2. F. Daniel, S. Soi, S. Tranquillini, F. Casati, C. Heng, L. Yan. From People to Services to UI: Distributed Orchestration of User Interfaces. *BPM'10*, Hoboken, NJ, USA.
3. OASIS. Web Services Business Process Execution Language Version 2.0, April 2007. [Online]. *http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html*
4. Active Endpoints Inc., Adobe Systems Inc., BEA Systems Inc., International Business Machines Corporation, Oracle Inc., SAP AG. WS-BPEL Extension for People (BPEL4People), Version 1.0. June 2007.
5. R. Acerbis, A. Bongio, M. Brambilla, S. Butti, S. Ceri, P. Fraternali. Web Applications Design and Development with WebML and WebRatio 5.0. *TOOLS'08*, pp. 392-411.
6. J. Gómez, A. Bia, A. Parraga. Tool Support for Model-Driven Development of Web Applications, *WISE'05*, pp. 721-730.
7. Sun Microsystems. JSR-000168 Portlet Specification, October 2003. [Online]. *http://jcp.org/aboutJava/communityprocess/final/jsr168/*
8. F. Daniel, F. Casati, B. Benatallah, M.-C. Shan. Hosted Universal Composition: Models, Languages and Infrastructure in mashArt. *ER'09*, pp. 428-443.

Smart Process Management: Automated Generation of Adaptive Cases based on Intelligent Planning Technologies[*]

Arturo González-Ferrer, Juan Fdez-Olivares, Inmaculada Sánchez-Garzón, and Luis Castillo

Department of Computer Science and Artificial Intelligence, Universidad de Granada,
{arturogf,faro,isanchez,l.castillo}@decsai.ugr.es

Abstract. This paper presents a proposal for the seamless integration of Intelligent Planning techniques into the life-cycle of BPM. The integration is intended to leverage current BPM techniques by allowing them to manage smart processes as adaptive business cases that can be automatically generated from original process models and executed in standard BPM runtime engines. The integration of such intelligent techniques is based on a two-fold transformation process: from business models into planning domains, and from plan representations into executable processes.

1 Motivation

Adaptive Case Management [15] is being used by the Workflow Management Coalition as the brand new name of an emergent paradigm in current BPM standard aimed at supporting Human-Centric processes[5] for knowledge workers (highly qualified personnel of organizations, like experts or decision makers). The processes required by knowledge workers are collections of tasks, which usually are collaboratively performed and which necessarily require human interaction in order to control and manage their execution. Such processes commonly support decisions and help to the accomplishment of workflow tasks of knowledge workers in several application domains. For the sake of simplicity, we will designate these processes as *Smart Processes*.

Smart Processes may be viewed as business cases that demand some kind of intelligent management [15] since, on the one hand they are very difficult to foresee, as they respond to *unstructured* sets of procedures which reside either in experts'mind or in documents, what makes difficult to devise a priori which tasks to execute. On the other hand, they need to be adaptively generated, since they are unpredictable and they strongly *depend on the context* of the organization and *do not respond to a fixed pattern*. Finally, they have to be *flexibly and interactively executed* by humans since they are subject to change.

There is a general consensus on that BPM technologies should be improved in order to support this kind of processes since, at the time being, BPM is mainly focused on

---

the management of static, repetitive, even perfectly predictable tasks/processes, mostly devoted to low qualification operators[15]. This is a widely known weakness and, because of this, it is also recognized that new techniques must be developed at both steps, process modeling/generation and process execution, in order to fully cover the needs of knowledge workers on Smart Processes.

In this sense, we present in this work a proposal that leverages the current BPM life-cycle in order to support smart processes through the development of Knowledge Engineering and intelligent planning techniques, focused on a two-fold transformation process. On the one hand, a transformation from business models into planning domains, in order to make the output of a business modeling tool interpretable by an intelligent planner. On the other hand, a transformation from a plan representation into an executable process, in order to make the output of the planner understandable by a BPM runtime engine.

The reason for the first transformation process is the fact that Artificial Intelligent Planning and Scheduling [10] AIP&S is a technology that clearly covers the above exposed demands for smart processes. Concretely, the hierarchical planning paradigm (mostly based on HTN, Hierarchical Task Networks [14,4]) has been proven in many applications([6,3,2,7]) to be successful on supporting the knowledge workers' effort. On the one hand, by modeling expert knowledge with planning domain models (which allow the description of actuation protocols or operating procedures represented as a hierarchy of tasks networks [13,4]). On the other hand, helping them to adaptively produce plans to support their decisions, as the result of a knowledge-driven planning process, guided by the knowledge represented in the planning domain. The reason for the second transformation process is that BPM has demonstrated to be much more appropriate for supporting the execution of the result of knowledge workers' effort, by providing technological infrastructures in order to interactively execute and monitor processes. Therefore, translating a generated plan into a BPM executable format will allow to execute plans on already existing standard platforms. Both transformations are fully automated, what allows to seamlessly integrate these techniques into the current BPM life-cycle, leading to an integrated framework for *Smart Process Management* that supports the automated generation of adaptive cases, from an original business process model based on AI P&S techniques.

Next sections are devoted to briefly explain the most relevant aspects of the framework as well as its main advantages.

2 Integrating Intelligent Planning into the BPM life-cycle

AI Planning and Scheduling [10] and more concretely HTN planning [4,13] becomes the central technique for this work since it supports the modeling of planning domains in very similar terms to the ones used in standard BPM models. An HTN domain is a compositional hierarchy of tasks networks representing activities at different levels of abstraction (either compound or primitive tasks). A domain describes how every compound task may be decomposed into (compound/primitive) sub-tasks and the order that they must follow, by using different methods. An HTN planner interprets the set of task decomposition schemes and reasons about them in order to compose a suitable plan

(a set of activities subject to order and temporal relations) such that its execution reaches a given goal, starting from an initial state (that represents an initial context as initial values for the properties of objects or resources involved in the activities). The HTN reasoning process is a knowledge and goal-driven process, guided by the procedural knowledge encoded in the domain. HTN techniques have been recently enhanced with valuable temporal and resource reasoning extensions [4], what allows to cope with a very rich temporal and resource representation, as well as to obtain plans that could be flexibly executed since they contain temporal constraints that can be adapted during plan execution.

From the AI P&S point of view, the need to obtain a context dependent, concrete workflow from a given business process model can be seen as the problem of obtaining a situated plan from (1) a planning domain that represents the original process model and (2) from an initial state that represents the context for which the business case has to be adapted. This aspect is the cornerstone of the proposal here presented, called Jabbah, a Knowledge Engineering for Planning tool that supports a three-step process that starts on an initial, already defined business process model, represented in XPDL using standard BPM modeling techniques. At a first step, the XPDL-based process model is transformed into an HTN planning domain and problem. Second, an HTN planner taking as input the domain and the problem (representing the context under which the case has to be enacted), generates a situated plan that represents the case to be executed. Hence, by using Jabbah in order to generate HTN domain and problem files, from an original process model, it is possible to carry out a knowledge-driven HTN planning process that results in the generation of situated plans, that is, plans customized for a given situation. These plans can be used either for supporting decision making about activity planning or process validation based on use-case analysis, leveraging the current BPM life-cycle at its process modeling/generation step. Third, the plan is finally transformed into a process in an executable format (XPDL again), and this process is then used as input to a standard BPM runtime engine in charge of supporting its interactive, human-centered execution. Details about these steps are explained in the following.

Transformation from process models to planning domains. Given an XPDL process as input (which can be clearly seen as a graph), Jabbah proceeds by identifying common workflow patterns (that is, sequential, parallel, subprocess and conditional structures) as process blocks in the process model, and then generate a tree-like structure, much similar to HTN domains. The HTN target domain language (called HTN-PDDL) used in this work is a temporally extended, hierarchical extension of PDDL [9], the standard language of planning domains (see [4,7] for details about this representation). Concerning the representation of preconditions and effects found in planning representations, we only deal at the moment with the conditions that have been defined in the original BPMN model, as well as the task order established on it. However, the BPMN notation allows the inclusion of customized annotations, that could be used to augment the knowledge about preconditions and effects present on the process model (i.e. by using *extendedAttributes* for activity nodes). This said, the Knowledge Engineering process for transforming process models into planning domains consists of three different stages: **i)** Firstly, the XPDL document is parsed, transforming it into an intermediate data structure and graph model that can be easily managed throughout the

next stages. **ii)** Then, the different blocks of workflow patterns (serial, parallel, subprocess and conditional blocks) are detected, distinguishing their kind from the knowledge acquired in the previous parsing stage, and build up an equivalent tree-like model. To do this, a collateral challenge is the transformation of the graph into a tree-like structure, which has been done using an algorithm described in [1]. This is carried on by arranging those workflow patterns hierarchically, but also keeping the semantic information (about control flow and decisions) present in the process diagram (see [11] for more details). c) Finally, we need to do a planning language generation phase, where we analyze the tree model that has been populated previously, trying to generalize common patterns found in the graph (i.e. serial or parallel split-joins patterns are always coded in the same way), and writing the HTN-PDDL code that corresponds to the tree-graph fragment analyzed.
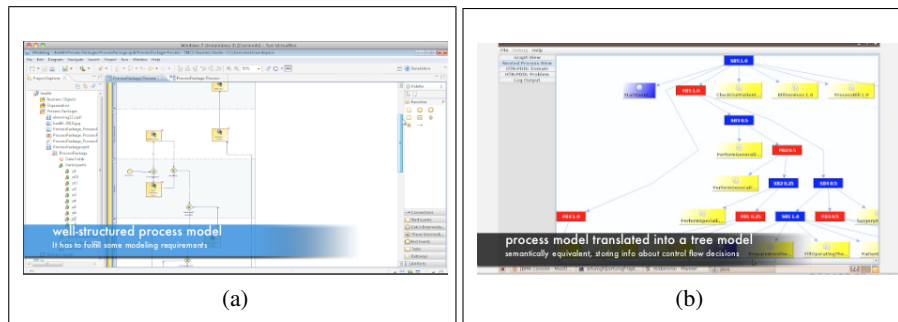

Fig. 1: A well-structured process model is designed with a BPM suite (a), and transformed later into a corresponding tree model by Jabbah (b), more appropriate to extract the HTN planning domain

By following this process, it is possible to generate domain and problem files which are given as input to an state-of-art HTN planner in order to obtain situated plans. We have used the IACTIVE™ planner for this work, a temporally extended HTN planner which uses HTN-PDDL as its planning language. Moreover, it has already been used in several applications [3,7,6]. These plans are generated by the planner for a given context represented in the problem file, and they can be interpreted as adaptive business cases since they are direct and automatically obtained from the initial process model. Given that the context parameters that guide the deliberative reasoning of the planning stage are included in the problem file, dynamic changes on the environment should be monitored, modifying the problem file accordingly, triggering a replanning stage to generate a new situated plan (the domain file would not be modified, in order to respect the original process model). Some approaches already exists for the monitored execution of plans [12], in order to handle any kind of exogenous events. The design of such execution monitor for the IACTIVE planner is being carried out at the moment of writing this paper.

Next, we briefly describe how these plans are transformed back into XPDL process instances in order to be understandable, and so executable, by a BPM engine.

Transformation from plans to executable process models. Given an XPDL process instance as input, BPM engines are commonly endowed with the necessary machinery in order to interactively execute every task in the process (allowing to start, finish, suspend or abort it) by following an execution model based on state-based automata. The plans generated by the planner, using the planning domains and problems generated by Jabbah, are represented in XML as a collection of *Task nodes* where every node contains information about: *actions (activities)* and their parameters; temporal information as *earliest start and earliest end* dates for the execution of every activity; *order dependencies* between actions which allow to establish sequential and parallel runtime control structures; and *metadata* which allow to represent additional knowledge like the user-friendly *description* of a task, its *type* (manual, auto) or its *performer* (that is, the participant of the activity). It is worth to note that metadata are generated at domain generation phase and are automatically extracted and generated by Jabbah. Starting from this XML plan representation, we have implemented as an extension of Jabbah a translation process that automatically generates XPDL processes which can be directly executed in a BPM runtime engine and users can interact with them on an underlying BPM console (see [8] for more details). This process has three main steps: (1) generation of XPDL DataFields and Participants from the problem and domain files; (2) generation of XPDL activities from the information about actions, temporal constraints and metadata in the plan; (3) generation of XPDL transitions from the order dependencies between the actions of the plan.

3 Conclusions

Jabbah fulfills, by using AI P&S knowledge engineering techniques, the needs of knowledge workers not yet completely covered by BPM technologies, in the management of dynamic, adaptable processes. The main innovative aspects of this framework are both, the fully automated transformation from a business process model (represented in XPDL) into an HTN planning domain and the translation from plan representation into an executable process format, what allows to directly execute the result of a planning process in standard BPM runtime engines. The framework described presents significant advances in the field of BPM, since the seamless integration of the above explained techniques leverages the BPM life-cycle, allowing it to automatically carry out adaptive case generation (starting from an initial process model represented in BPM standard languages), and to execute cases using standard BPM technologies.

To date, JABBAH has been tested in two case studies drawn from different domains, e-learning and e-health. The first model represents the whole process to develop and deploy a specific course within the e-learning center at the University of Granada. So, having an incoming course request, as well as some available workers with different capabilities each, we want to assign an activity to every worker, so that a plan over time can be obtained, providing the e-learning managers information that helps to do anticipated decision-making upon the course request. The second one represents a general care-process starting from a patient admitted into the hospital and finishing when the

health insurance billing for this patient takes place. In this second process, we can better observe how process planning is carried out, given different input parameters which can vary in real situations (e.g. is it an emergency? does it need an urgent operation?) and how our tool is able to generate different process instances according to them.

Source code and details about the Jabbah framework and the case studies commented are available at its website[1]. A demonstration screencast about its operation can be watched in the "Screenshots" subsection of the website, or directly in Youtube[2].

References

1. Bae, J., Bae, H., Kang, S., Y.Kim: "Automatic Control of Workflow Processes Using ECA Rules". IEEE Transactions on Knowlegde and Data Engineering 16(8) (2004)
2. Bresina, J.L., Jonsson, A.K., Morris, P., Rajan, K.: Activity planning for the mars exploration rovers. In: Proceedings of the ICAPS05. pp. 40–49 (2005)
3. Castillo, L., Fdez-Olivares, J., García-Pérez, O., Garzón, T., Palao, F.: Reducing the impact of ai planning on end users. In: ICAPS 2007, Workshop on Moving Planning and Scheduling Systems into the Real World. pp. 40–49 (2007)
4. Castillo, L., Fdez-Olivares, J., García-Pérez, O., Palao, F.: Efficiently handling temporal knowledge in an HTN planner. In: Proceeding of ICAPS06. pp. 63–72 (2006)
5. Dayal, U., Hsu, M., Ladin, R.: Business process coordination: State of the art, trends, and open issues. In: Proceedings of the 27th VLDB Conference (2001)
6. Fdez-Olivares, J., Castillo, L., García-Pérez, O., Palao, F.: Bringing users and planning technology together. Experiences in SIADEX. In: Proceedings ICAPS06. pp. 11–20 (2006)
7. Fdez-Olivares, J., Castillo, L., Cozar, J., Garcia-Perez, O.: Supporting clinical processes and decisions by hierarchical planning and scheduling. Computational Intelligence To Appear (2010)
8. Fdez-Olivares, J., González-Ferrer, A., Sanchez-Garzón, I., Castillo, L.: Integrating plans into BPM technologies for human-centric process execution. In: ICAPS 2010. Proceedings of Workshop on Knowledge Engineering for Planning and Scheculing (KEPS) (2010)
9. Fox, M., Long, D.: PDDL2-1: an extension to PDDL for expressing temporal planning domains. Tech. rep., University of Durham, UK (2001)
10. Ghallab, M., Nau, D., Traverso, P.: Automated Planning. Theory and Practice. Morgan Kaufmann (2006)
11. González-Ferrer, A., Fdez-Olivares, J., Castillo, L.: "JABBAH: A Java Aplication Framework for the Translation between Business Process Models and HTN". In: Proceedings of ICKEPS Competition (2009)
12. Leoni, M.D.: Adaptive process management in highly dynamic and pervasive scenarios. In: YR-SOC 2009. pp. 83–97
13. Nau, D., Au, T., Ilghami, O., Kuter, U., Murdock, J.W., Wu, D., Yaman, F.: SHOP2: An HTN Planning System. Journal of Artificial Intelligence Research 20, 379–404 (2003)
14. Sacerdoti, E.D.: The nonlinear nature of plans. In: Proceedings of IJCAI 1975. pp. 206–214 (1975)
15. WfMC: Group on adaptive case management (2010), `http://www.xpdl.org/nugen/p/adaptive-case-management/public.htm`

---

[1] `http://sites.google.com/site/bpm2hth`

[2] `http://www.youtube.com/watch?v=FOHYsMWvS1c`

ProM 6: The Process Mining Toolkit

H.M.W. Verbeek, J.C.A.M. Buijs, B.F. van Dongen, W.M.P. van der Aalst

Department of Mathematics and Computer Science,
Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{h.m.w.verbeek,j.c.a.m.buijs,b.f.v.dongen,w.m.p.v.d.aalst}@tue.nl

Abstract. Process mining has been around for a decade, and it has proven to be a very fertile and successful research field. Part of this success can be contributed to the ProM tool, which combines most of the existing process mining techniques as plug-ins in a single tool. ProM 6 removes many limitations that existed in the previous versions, in particular with respect to the tight integration between the tool and the GUI.
ProM 6 has been developed from scratch and uses a completely redesigned architecture. The changes were driven by many real-life applications and new insights into the design of process analysis software. Furthermore, the introduction of XESame in this toolkit allows for the conversion of logs to the ProM native format without programming.

1 Introduction

Process mining allows for the extraction information from event logs [2, 3, 5, 8], and is closely related to BAM (Business Activity Monitoring), BOM (Business Operations Management), BPI (Business Process Intelligence), and data/workflow mining. Unlike classical data mining techniques the focus is on processes and questions that transcend the simple performance-related queries supported by tools such as Business Objects, Cognos BI, and Hyperion.

ProM is a generic open-source framework for implementing process mining algorithms in a standard environment [6, 1]. ProM 5.2 features over 280 plug-ins for process mining, analysis, monitoring and conversion. ProM is available as binary distribution files for Windows, Mac OS X and Unix platforms, and as source code under the terms of the CPL license.

Although ProM (up to version 5.2) has been a huge success, it did have a number of limitations. First of all, the plug-in concept lacked a clean separation between its actual process mining algorithm and its GUI. As a result, one could only use an algorithm in a context that allowed the user to provide necessary parameters through a GUI. When process mining applications become more mature/challenging, this limitation is not acceptable anymore.

A second limitation was the fact that the framework was unaware of the inputs required by the plug-ins and the outputs they delivered (for example, plug-ins were not explicitly annotated with type information to allow chaining of plug-ins). As a result, it was not really possible to define 'macro' plug-ins. The
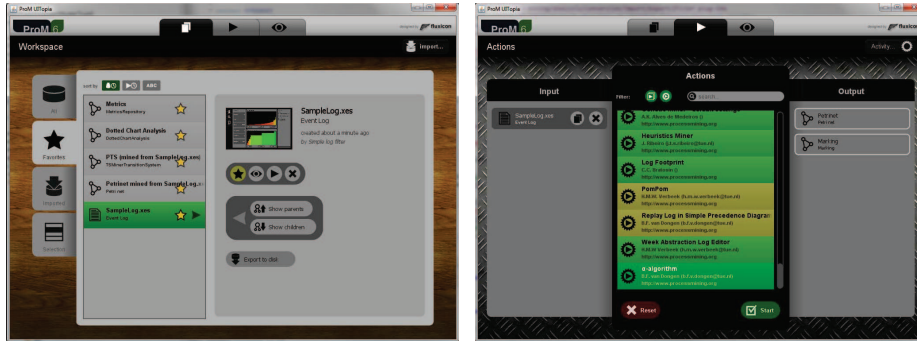
Fig. 1. ProM 6 Look&Feel

framework simply did not know which plug-ins could be 'chained' into a single plug-in. A third shortcoming were problems related to licenses. As an example, the often-used Decision Point Analysis plug-in could not be added to a ProM release because it used an L-GPL licensed library, whereas the ProM framework used the CPL license.

This paper introduces the new version of ProM, ProM 6, which is a complete overhaul of the former ProM versions that alleviates the shortcomings mentioned above. Unfortunately, as a result, plug-ins from versions 5.2 and earlier cannot run in ProM 6. However, as this paper shows, several plug-ins have been re-implemented. Moreover, recent research results are available as ProM 6 plug-ins (and not in ProM 5.2). Along with ProM, the ProM Import Framework has also been completely redesigned with the ease of defining new conversions from legacy log formats to the XES format in mind. The resulting tool called XESame is included in the ProM 6 toolkit.

2 ProM 6

The left-hand side of Fig. 1 shows the workspace of ProM 6. This workspace shows the object pool (containing logs, process models, . . .), and the actions the user can take on a selected object. In Fig. 1, a log called "SampleLog.xes" has been selected, which was obtained from another log by filtering it using the "Simple log filter". Apart from other things, this view also offers the possibility to import an object from file, and to export an object to file.

The right-hand side of Fig. 1 shows the action view for the log object mentioned earlier. This view shows a list of possible actions (plug-ins) together with the required inputs and expected outputs of the selected action. The color of the action indicates whether all (green), some (yellow), or none (red) of the required inputs are present and meet the conditions. In Fig. 1, the "α-algorithm" has been selected on the "ExampleLog.xes" log, which requires an "Event log" as input and is expected to output a "Petri net" and a "Marking" (the initial

marking of the Petri net). As all inputs of the "α-algorithm" are present (it is green), it can be run immediately by selecting the "Start" button.

2.1 Contexts

In earlier versions of ProM, the actual process mining algorithms implemented by plug-ins assumed the presence of a GUI. Most algorithms require parameters, and the plug-in would ask the user for these parameters using some GUI-based dialog. Furthermore, some plug-ins displayed status information using progress bars and such. Thus, the actual process mining algorithm and the use of the GUI were intertwined. As a result, the algorithm could only be run in a GUI-aware context, say on a local workstation. This way, it was impossible to effectively run process mining experiments using a distributed infrastructure and/or in batch.

In ProM 6, the process mining algorithm and the GUI have been carefully separated, and the concepts of *contexts* has been introduced. For a plug-in, the context is the proxy for its environment, and the context determines what the plug-in can do in its environment. A plug-in can only display a dialog or a progress bar on the display if the context is GUI-aware. Typically, in ProM 6 a plug-in is split into a number of plug-ins: A plug-in for every context. The actual process mining algorithm will be implemented in a generic way, such that it can run in a general (GUI-unaware) context. This allows the algorithm to be run in any context, even in a distributed context. The dialog for setting the required parameters is typically implemented in a GUI-aware variant of the plug-in. Typically, this GUI-aware plug-in first displays the parameter dialog, and when the user has provided the parameters and has closed the dialog, it will simply run the generic plug-in using the provided parameters.

The major advantage of this is that the ProM framework may decide to have the generic plug-in run on a different computer than the local workstation. Some plug-ins may require lots of resources (time, memory, disk, . . .), like for example the genetic miner. Basically, the genetic miner takes a model and a log, and then generates a number of alternative models for the given log. The best of these alternatives models are then taken as new starting points for the genetic miner. The genetic miner repeats this until some stop criterium has been reached, after which it returns the best model found so far. Clearly, this miner might take considerable time (it may take hundreds of iterations before it stops and the fitness calculation is very time-consuming for large logs), and it may take considerable memory (the number of alternative models may grow rapidly). For such an algorithm, it might be preferable to have it run on a server which is more powerful than the local workstation. Moreover, genetic mining can be distributed in several ways. For example, the population can be partitioned over various nodes. Each subpopulation on a node evolves independently for some time after which the nodes exchange individuals. Similarly, the event logs may be portioned over nodes thus speeding up the fitness calculations.

2.2 Chaining

In earlier versions of ProM, the framework did not know the required inputs and/or the expected outputs of a plug-in. As a result, it was impossible to refer to a required input or an expected output. Hence, it was not possible to combine a number of plug-ins into a 'macro' plug-in.

In ProM 6, the framework can tell the what the required inputs for a given plug-in are and what the expected outputs are. See also the action view (right-hand side of Fig. 1), which shows the inputs and outputs for the "α-algorithm" plug-in. By selecting a plug-in, the action view will show the required inputs and the expected outputs for the selected plug-in. By selecting appropriate objects as inputs, the plug-in can then be run. Furthermore, the framework can also tell which plug-ins fit with given a set of input types and a given set of output types. Suppose we have an event log, and we want to create a Petri net model from this log. Then we can simply select the "Event log" type as input for the plug-in and the "Petri net" type as expected output, and the action view will show us the list of plug-ins that actually take an event log as input and produce a Petri net as output.

Another advantage of this feature is that we can now delegate the construction of an object of a given type out of a collection of existing objects, without specifying which plug-in should be used, to the framework. The framework simply takes the existing objects as inputs, the given object type as the output type, and searches for the plug-ins that fit these inputs and outputs. Out of these fitting plug-ins one is selected and run to obtain the object of the given type. This way, a plug-in can run another plug-in without knowing that other plug-in. Furthermore, if a third plug-in becomes available that is superior to the other plug-in, we can simply replace the other plug-in by the superior plug-in.

2.3 Packages

In earlier versions of ProM, the collection of supported plug-ins was fixed. As a result, plug-ins with conflicting licenses could not be added to any ProM distribution, and the distribution most likely contained a number of plug-ins that were of no use to the user. As a result of the former, useful plug-ins like the "Decision Point Analysis" had to be left out of the standard ProM distribution, as ProM itself used the CPL license whereas the plug-in used an L-GPL licensed library. Besides licence issues there were other reasons for changing the management of plug-ins. For example, some users were overwhelmed by the many plug-ins (over 200) signaling the need for versions with smaller subsets of plug-ins.

In ProM 6, plug-ins can be installed in a dynamic way using the *ProM Package Manager*. The ProM Package Manager is a separate tool that allows the user to add and/or remove so-called packages to the installed ProM distribution. Every package contains a collection of plug-ins, and these plug-ins are installed if the package has been installed. As a result, ProM 6 can be distributed with only a core set of plug-ins, and users can add relevant plug-ins themselves. Furthermore, plug-in developers can create their own packages using only the ProM
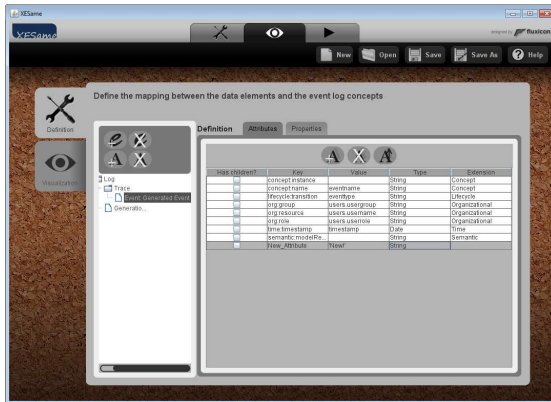
Fig. 2. XESame screenshot

distribution: No ProM sources are required to create packages. Finally, plug-in developers can create their own package repositories, and users can install these packages through the Package Manager.

3 XESame

Although many information systems record the information required for process mining, chances are that this information is not readily available in an event log format. In ProM 5.2, the ProM Import Framework [7] was included in the ProM 5.2 for converting logs to MXML. Although there is a collection of plug-ins for various systems and data structures, new plug-ins often needed to be written in Java for a new process mining project. The main problem with this approach is that one cannot expect a domain expert to have Java programming skills. Therefore, *XESame* [4][1] has been developed and is included in the ProM 6 toolkit. XESame allows a domain expert to extract the event log from the information system at hand without having to program.

Fig. 2 shows the main conversion definition screen of XESame. Here the user can set the attribute values for different elements of the XES event log as shown in the tree at the left hand side. In this example attributes for an event are defined as displayed at the right hand side. Most of the attribute values are extracted from the data source. The name of the event, which is recorded in the *concept:name* attribute, for instance is extracted from the *eventname* field in the data source. The name of the user that executed the event is recorded in the *org:resource* attribute and is extracted from the *username* field of the related record in the *users* table. Furthermore, a custom attribute *New_Attribute* was defined which has the fixed value *New!* for each event in the resulting event log.

---

[1] Note that in [4] the tool is called the XES Mapper instead of XESame

4 Conclusions

ProM 6 is not just a upgrade from ProM 5.2. The tool has been completely re-implemented using an enhanced architecture and user interface. It offers many features that were not possible in earlier versions of ProM:

– Plug-ins can run in distributed environments,
– Plug-ins can be chained into 'macro' plug-ins,
– Plug-ins can be installed and updated at run-time,
– Logs can be extracted from information systems without the need for programming.

Because of the new architecture, the plug-ins from the earlier versions do not run under ProM 6, which makes it necessary to re-implement them. We believe that the new features offered by ProM 6 are well worth the effort. Moreover, we also use the opportunity to improve the functionality of plug-ins based on lessons learned for many real-life applications.

References

1. W.M.P. van der Aalst, B.F. van Dongen, C.W. Günther, R.S. Mans, A.K. Alves de Medeiros, A. Rozinat, V. Rubin, M. Song, H.M.W. Verbeek, and A.J.M.M. Weijters. ProM 4.0: Comprehensive Support for Real Process Analysis. In J. Kleijn and A. Yakovlev, editors, *Application and Theory of Petri Nets and Other Models of Concurrency (ICATPN 2007)*, volume 4546 of *Lecture Notes in Computer Science*, pages 484–494. Springer-Verlag, Berlin, 2007.
2. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
3. R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process Models from Workflow Logs. In *Sixth International Conference on Extending Database Technology*, pages 469–483, 1998.
4. J.C.A.M. Buijs. Mapping Data Sources to XES in a Generic Way. Master's thesis, Eindhoven University of Technology, 2010.
5. A. Datta. Automating the Discovery of As-Is Business Process Models: Probabilistic and Algorithmic Approaches. *Information Systems Research*, 9(3):275–301, 1998.
6. B.F. van Dongen, A.K. Alves de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W.M.P. van der Aalst. The ProM framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer-Verlag, Berlin, 2005.
7. C. Günther and W.M.P. van der Aalst. A Generic Import Framework for Process Event Logs. In J. Eder and S. Dustdar, editors, *Business Process Management Workshops, Workshop on Business Process Intelligence (BPI 2006)*, volume 4103 of *Lecture Notes in Computer Science*, pages 81–92. Springer-Verlag, Berlin, 2006.
8. J. Herbst. A Machine Learning Approach to Workflow Management. In *Proceedings 11th European Conference on Machine Learning*, volume 1810 of *Lecture Notes in Computer Science*, pages 183–194. Springer-Verlag, Berlin, 2000.

Business Process Modeling and Quick Prototyping with WebRatio BPM

Marco Brambilla[1], Stefano Butti[2], Piero Fraternali[1]

[1] Politecnico di Milano, Dipartimento di Elettronica e Informazione
P.za L. Da Vinci, 32. I-20133 Milano - Italy
{`marco.brambilla, piero.fraternali`}`@polimi.it`
[2] Web Models S.r.l., I-22100 Como - Italy
`stefano.butti@webratio.com`

Abstract. We describe a software tool called WebRatio BPM that helps close the gap between the modeling of business processes and the design and implementation of the software applications that support their enactment. The main idea is to enhance the degree of automation in the conversion of business process models into application models, defined as abstract, platform-independent representations of the application structure and behavior. Application models are themselves amenable to the semi-automatic transformation into application code, resulting in extremely rapid prototyping and shorter time-to-market. Thanks to the proposed chain of model transformations it is also possible to fine tune the final application in several ways, e.g., by integrating the visual identity of the organization or connecting the business process to legacy applications via Web Services.

1 Introduction

Business process modeling has become the recognized best practice for enterprise-wide application specification. Business process languages and execution environments ease the definition and enactment of the business constraints, by orchestrating the activities of employees and of computer-supported services.

However, turning a business process model into the specification, design and implementation of a software solution for process enactment is a non trivial task: the specified processes can be a mix on new functionality to be developed and interactions with pre-existing systems and the user's activities must be supported through effective and usable interfaces, possibly compliant with the visual identity and interaction style of other corporate applications. Furthermore, the business requirements embodied in the process models, as well as the technical context in which the underlying applications are deployed, are subject to evolution. This may cause severe alignment problems when trying to keep the business process and the application in sync.

The gap between process modeling and application development can be alleviated by increasing the degree of automation in the design, implementation and maintenance of applications derived from process models. The automation
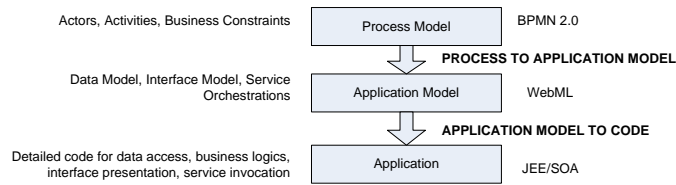
Fig. 1: Overview of the models and transformations

framework should support the semi-automatic translation of the process model into running applications, be flexible enough to incorporate different architectural and interaction requirements, and apply also to application evolution and maintenance. An outstanding difficulty is the semantic distance between the process model and the the running application: the former dictates roles, activities and business constraints at a very abstract level, irrespective of how these are supported by computing tools; the latter embodies very low-level details, where business, architecture, and interaction aspects are blended together and hard to customize and evolve separately. As an example, the same user's activity specified in the process model could be enacted in a variety of ways: by means of a wizard, by form editing, by using a legacy application interface, and so on. This observation motivates the approach applied in this paper, which is based on the three level conceptual architecture illustrated in Figure 1.

Following the approach of Model Driven Engineering, the business requirements and the application are represented using models, organized at three levels:

The models managed by our approach are organized in three levels: (i) the business model (specified with BPMN [6]); (ii) the structure and behavior of the application (expressed in WebML [3]); and (iii) the executable application running code. Application development is then seen as two consecutive transformations: the *Process Model to Application Model transformation*, and the *Application Model to Running Code transformation*.

The introduction of the application modeling layer increases the complexity of the conceptual architecture, but brings fundamental advantages: there is one place (the application model), where it is possible to reason about the distinct aspects of the application separately; the BPM to Application transformation can be supplied with transformation rules capable of producing alternative ways of encoding an activity, by using different patterns; automatically generated application models can be fine tuned, to introduce usability patterns, without breaking the application compliance to the process model; application evolution can be performed independently of the technical platform, by updating the application model and then regenerating the application code.

In the scientific community, several works have addressed the binding of BPM and Model Driven Development of Web applications: PML [5], YAWL [4], OOHDM [7], WSDM [8] and others. Our previous work [2] established the theoretical basis of the implementation described in here; with respect to that early
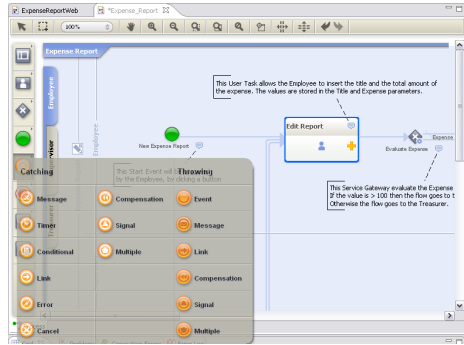
Fig. 2: The new WebRatio BPMN editor interface.

idea, now the BP model and the application model are treated as orthogonal and independent models.

Based on these premises, our original contributions are: (i) a model-driven perspective to business process-based software application development, which leverages the integrated use of two orthogonal models (BP and application models); (ii) a generative framework for producing the executable code from the process and application model, and a one-click, zero-coding generation of a running prototype on an enterprise class standard architecture; (iii) the implementation of the model editor (Figure 2) and transformations in a commercial tool suite called WebRatio [1], which supports all the steps of the proposed approach.

2 Models and Transformations

For describing the BP model, we adopt BPMN 1.2., plus some features of BPMN 2.0, whose Beta specification is currently available. Some features of version 2.0 of the language are of extreme importance for the generation of application models, namely: *DataInputs*, *DataOutputs*, and *DataInputAssociations*, which explicitly specify the inputs and outputs of executable tasks; the classification of *Tasks* in *UserTasks* and *ServiceTasks*; and others.

As a second level of modeling, we propose the application model, aiming at describing all the user interaction, service calls, and navigational aspects of the designed application. We focus on the Web as the platform of choice for the implementation of the software application, which is in line with the current trends in enterprise application development. Therefore, we adopt the *WebML* notation [3], a visual language for designing data- and service-centric Web applications [2], that allows specifying the conceptual model of applications built on top of a data schema and composed of one or more hypertexts used to publish or manipulate data. Upon the same data model, different hypertext models (*site views*) can be defined (e.g., for different types of users or devices). A site view is a graph of *pages*, consisting of connected *units*, representing data publishing

components. Units are related to each other through *links*, representing navigational paths and carrying parameters. Starting from a BPMN specification, an automatic transformation produces the logical representation of the process metadata and a WebML Application Model, comprising a Data Model dictating the application-specific concepts and a collection of Site Views and Service Views, including the primitives for the user interaction and Web service orchestrations.

Process Data Model Generation. The transformation from BPMN to the process data model consists of an encoding of the BPMN concepts in a relational database structure: the BPMN precedence constraints and gateways are transformed into instances of a relational representation. At runtime, the BPMN constraints, stored in the Process Metadata Schema, are exploited by the components of the Application Model for enacting the precedences among human-executed tasks and executing the service invocations.

Application Model Generation. The transformation from BPMN to WebML considers the type (User or Service) of the gateways and of the tasks, as well as the information on the control and data flows. The generated application models consist of a coarse set of user interfaces and prototype business operations. Process control is encapsulated thanks to the automatically generated process data model: the computation of the next enabled activities given the current state of the workflow is delegated to a specific WebML component, called *Next* unit, which factors out the process control logic. The *Next* unit exploits the information stored in the process data model to determine the current process status and the enabled state transitions. The tool also automatically generates the WebML model for user-driven and automatic tasks (e.g., performed by web services) and the hypertext for managing the tasklist and the process execution status.

Figure 3a shows an excerpt of the WebML model automatically generated from BPMN, describing a hypotethical activity that allows users to submit leasing requests online; the module contains an *Input* component (1) for initializing the possible data values in input to the activity (e.g., *ContractID*, *ProductType*); it includes a page (*Customer Leasing Request* (2)) with a sub-module publishing information about the activity and the associated process execution (*Info* (3)), a component fetching the current values of the parameters (*GetInput* (4)) used to preload the fields of a form for submitting the user inputs (*UserInput*(5)). From the input form, three links allow the user to close (6) , suspend (7) , and cancel (8) the activity. For example, when the user closes the activity, the parameter values are stored (by the *SetParameters* unit (9)) and the next activities are calculated (by the *NextActivity* unit (10)); the user interface automatically generated with a default graphical style is shown in Figure 3b.

Generated runtime architecture. The run-time architecture of the applications generated by WebRatio starting from the application model exploits a set of off-the-shelf components for organizing the business tier: *Smart service creation* for components, created upon request, cached, and reused across multiple requesters; *XML parsing and access* granted by standard parsing tools; and *Con-*
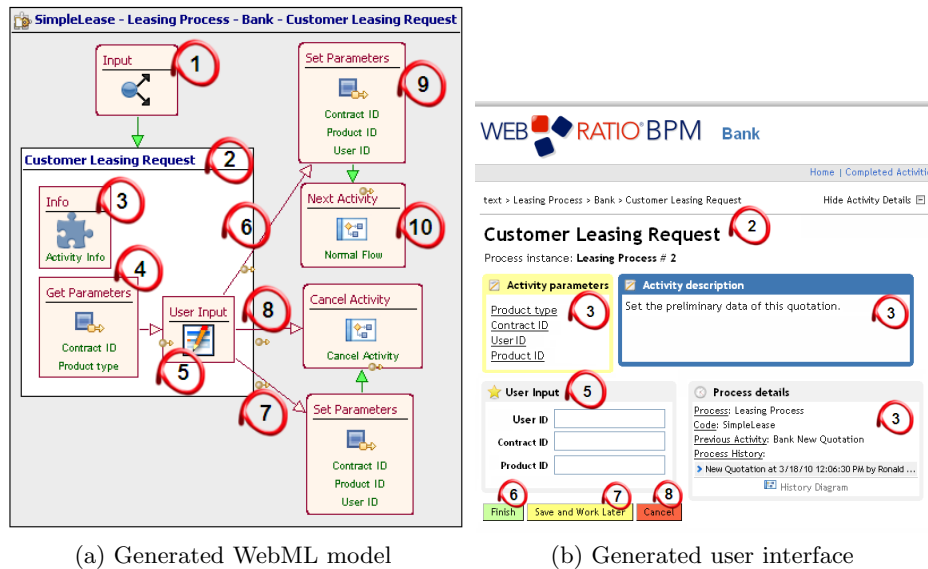
(a) Generated WebML model   (b) Generated user interface

Fig. 3: WebML model and user interface for Customer Leasing Request activity.

nection pooling for dynamically managing a pool of database connection allow to optimize performance and reliability. At runtime, only service component is deployed for each type of model concept and one runtime XML descriptor is deployed for each usage in the application model.

WebRatio supports also the new requirements imposed by Rich Internet Applications (RIAs), including complex user interactions such as drag and drop, partial page refresh, dynamic resizing of visual components, graphical editing of objects, in-page popup windows, splash screens, dynamic tooltips, and waiting animations, text autocompletion, on-event actions, and field dependencies.

3   Tool Implementation and Experience

Tool implementation. The proposed generative framework has been implemented as an extension of WebRatio, a Model-Driven Web application development tool allowing one to edit WebML models and automatically transform them into a running applications for JEE and Service Oriented Architectures. For supporting BPM design, the following extensions have been devised. The *model editing GUI* has been extended by with a new BPMN editor . The *code generator* has been extended with the transformation from BP model to application model; furthermore, the JEE code generation has been augmented to produce the relational instance of the Process Metadata and the Java code of novel WebML components (e.g., the Next unit). Moreover, a one-click publishing function has been added to the BPMN editor, thus allowing the immediate generation of a rapid prototype of the BPMN process. This functionality invokes

in sequence the two transformations from BPMN to WebML and from WebML to JEE, yielding a dynamic, multi-actor application with a default look & feel, as shown in Figure 3b. The generator creates a few exemplary users for each BPMN actor, which allows the analyst to impersonate each role in the process.

Industrial applications. The proposed method and tool have been tested in several real industrial scenarios: as an example, we report the outcomes of a project conducted within the leasing division of a major European bank, which needed to reshape its entire software architecture according to a BPM-based paradigm. The resulting pilot application covers 52 business processes, comprising more than 1,100 activities spanning 30 user roles. The development team comprised 14 staff members from 3 organizations, with total effort amounting to 2551 man days, spent in 375 elapsed days. Rapid prototyping let the team deploy 4 major prototypes and 35 minor releases along one year.

We estimated the spared effort induced by automatic model transformation by measuring the percentage of automatically generated model elements, which consists of a percentage ranging between 17% and 20%.

4   Conclusion

We presented a tool suite for supporting the model-driven integrated design of business processes and enterprise Web applications. Visual model design and model transformations allow designers to produce both as early prototypes and final applications without coding.

References

1. Roberto Acerbis, Aldo Bongio, Marco Brambilla, Stefano Butti, Stefano Ceri, and Piero Fraternali. Web applications design and development with webml and webratio 5.0. In *TOOLS (46)*, pages 392–411, 2008.
2. Marco Brambilla, Stefano Ceri, Piero Fraternali, and Ioana Manolescu. Process Modeling in Web Applications. *ACM TOSEM*, 15(4):360–409, 2006.
3. Stefano Ceri, Piero Fraternali, Aldo Bongio, Marco Brambilla, Sara Comai, and Maristella Matera. *Designing Data-Intensive Web Applications*. Morgan Kaufmann, USA, 2002.
4. A.H.M. Hofstede, W.M.P. van der Aalst, M. Adams, and N. Russell (Eds.). *Modern Business Process Automation, YAWL and its Support Environment*. Springer, 2010.
5. John Noll and Walt Scacchi. Specifying process-oriented hypertext for organizational computing. *J. Netw. Comput. Appl.*, 24(1):39–61, 2001.
6. OMG, BPMI. BPMN 1.2. Technical report, http://www.bpmn.org/, 2009.
7. Hans Albrecht Schmid and Gustavo Rossi. Modeling and designing processes in e-commerce applications. *IEEE Internet Computing*, 8(1):19–27, 2004.
8. Olga De Troyer and Sven Casteleyn. Modeling complex processes for web applications using wsdm. In *Ws. on Web Oriented Software Technology (IWWOST)*, pages 1–12, Oviedo, 2003.

bflow* Toolbox - an Open-Source Modeling Tool

Christian Böhme[1], Jörg Hartmann[1], Heiko Kern[1], Stefan Kühne[1], Ralf Laue[2],
Markus Nüttgens[3], Frank J. Rump[4], and Arian Storch[2]

[1] Business Information Systems
Computer Science Faculty, University of Leipzig, Germany
`cboehme|jhartmann|kuehne|kern@informatik.uni-leipzig.de`
[2] Chair of Applied Telematics / e-Business
Computer Science Faculty, University of Leipzig, Germany
`laue|storch@ebus.informatik.uni-leipzig.de`
[3] Chair of Information Systems
University of Hamburg, Germany
`markus.nuettgens@wiso.uni-hamburg.de`
[4] University of Applied Science Emden-Leer, Germany
`rump@informatik-emden.de`

Abstract. The *bflow* Toolbox* is a modeling tool for the business process modeling language Event Driven Process Chains (EPCs).
In this paper, we describe three innovative features of the modeling tool:
First, the modeler gets continuously feedback about possible modeling problems. Second, there is an option to construct a large part of a model with input from the keyboard only, i.e. without ever touching the mouse. And third, new features can be added to the tool very easily - without the need to be familiar with Eclipse development.

1 Introduction

The *bflow* Toolbox* is an open-source tool for graphical business process modeling in the Event-Driven Process Chains notation. Currently it supports three types of diagrams: Extended EPCs, Object-Oriented EPCs and Value Chain Diagrams. The first release has been published in March 2008 as a joint effort between developers at the University of Hamburg and the University of Applied Sciences Emden/Leer. Later, the University of Leipzig joined the developer team.

bflow Toolbox* is a plugin of Eclipse and makes use of the Eclipse Modeling Framework (EMF) and the Eclipse Graphical Modeling Framework (GMF). This means that it is based on a well designed, coded and tested code base. It makes use of the usual features provided by EMF and GMF like storing models as XMI-files, collapsing and expanding modeling elements, aligning modeling elements, using the clipboard, etc.

The adequacy of our tool for real-world usage has been shown by the city of Düsseldorf where the *bflow* Toolbox* has been used for modeling communal business processes in order to comply with the EU services directive.

In this paper, we will present three innovative features of the *bflow* Toolbox*. In Sect. 2, we will discuss the "continuous verification" feature. In Sect. 3, we

present a user interface that allows modeling structured model parts very quickly. And in Sect. 4, we show how we support developers who want to add features to the modeling tool without having to know implementation details.

2 Continuous Verification

Verification of business process models has been studied for a long time, and there are reliable and fast methods for deciding about important properties like soundness [1, 2]. However, the usual way of applying these methods is to analyze a model *after* it has already been completed.

Similar to techniques such as continuous compilation and continuous testing that are integrated into modern software development systems, we use the concept of *continuous verification*: Verification runs in background at the time of modeling. If a possible modeling problem is detected, the modeler is alerted immediately. Our tool also shows the locations of error causes in the visual representation of the model and suggests how to fix the problems.

The approach for locating possible modeling problems is based on a heuristic algorithm. It locates patterns that are usually related to a modeling problem. These patterns include syntactical problems as well as "technical" errors (such as deadlocks in the control flow). Another type of problem patterns can be used to detect parts of the model that can be regarded as "bad modeling style". In [3], we have shown that our heuristic approach identifies violations of the soundness property almost as accurate as model-checkers. While checking for syntactical correctness is included in some other tools, this is not the case for checking the correctness of the control flow and the modeling style.

With the help of our pattern-based algorithm we are also able to detect some commonly occurring errors that can only be identified by examining the labels of the functions and events. An example is shown in Fig. 1: Our verification algorithm complains if an AND-split is followed by two events such that the labels of those events contradict each other (i.e. the events cannot occur together).

With continuous verification, the modeler gets an immediate feedback about modeling problems, i.e. errors can be detected and fixed without delay. In [4], we have published the results of a case study where the presence of this feature helped novice modelers to decrease the number of (syntactical) errors in a model by 79%. More details about the continuous verification approach can be found in [5] and [3].
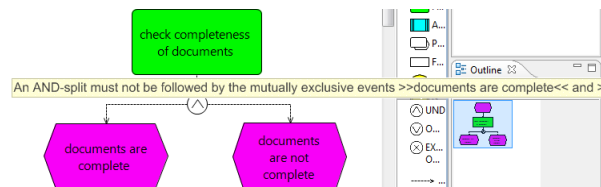

Fig. 1. Feedback on errors in the labels of the events

3 User Interface for Keyboard-Based modeling

Business process models (including EPCs) often contain large fragments of sequences (activities and events without routing elements between them) and well-structured control-flow blocks (like an AND-split/AND-join combination).

In the most visual modeling tools we are aware of, a modeler needs unnecessarily many steps to draw a rather simple model fragment like a sequence. The usual procedure for drawing just two activities which are connected by an arrow includes the following steps:

1. Select the shape for an activity from the menu
2. Click on the place where the shape should be located
3. Add a label
4. Repeat step 1-3 for the second activity
5. Select the shape for an arrow from the menu
6. Connect the two activities

Such a sequence of steps has to be repeated very often when a model is created. In the *bflow* Toolbox*, it is sufficient to enter the labels of the modeling elements into a table. By the click of a button, the visual representation of the activities described in the table is generated, i.e. the elements are added to the model. Fig. 2 shows the input for adding a simple structure between an exclusive choice and a simple merge; it will generate the model fragment shown in Fig. 3. Of course, the use of this wizard is optional; the user can always use the traditional click-and-arrange method.

While similar user interfaces have been described before [6, 7], tool vendors rarely adopt such functionality that can lead to saving a considerable part of modeling time.

4 Easy Extensibility

bflow* Toolbox has been developed as an open and extensible framework. We provide interfaces for adding new features to the *bflow* Toolbox*.
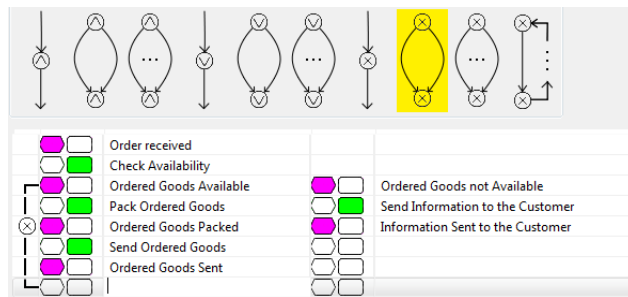

Fig. 2. User Interface for creating parts of the model very quickly
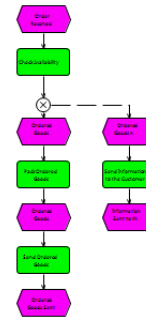
Fig. 3. Generated model fragment

If additional attributes are needed (for example for information about costs to an activity or for adding meta-information to an EPC as a whole), the *bflow* Toolbox* provides the possibility to add such attributes to a model (or to a group of models) at runtime. There is no need to change the EPC metamodel.

Using new export and import formats requires adding one or more XSLT transformation files and to insert information about the export/import format to a configuration file. This way, new exports/imports can be added without having to compile the *bflow* Toolbox* sources. In its current release, *bflow* Toolbox* comes with exports and imports for several other EPC modeling tools (ARIS, Semtalk, EPCTools and Oryx). For ARIS and Microsoft Visio, our tool also provides a metamodel-based model interchange. Details can be found in [8, 9].

Furthermore, we tried to make the integration of third-party programs into our tool as easy as possible. We know from our own experience that often a lot of knowledge is necessary before someone can actually do such integration. At least, the answers to the following questions have to be known:

- How can the model data be accessed?
- How can these data be transferred into the data format expected by the third-party program?
- How can we start the third-party program from within the modeling tool?
- How can we transfer the answers given by the third-party program back into the user interface of the modeling tool?

In the *bflow* Toolbox*, we provide easy-to-understand interfaces for dealing with the above questions. The already mentioned export scripts can be used for exporting the model data into the expected format. This means that there is no need to know anything about the internal architecture or the file format of the *bflow* Toolbox* for getting access to the data of the model.

The information on how to start the external program can be added to a configuration table (see Fig. 4) at runtime. If the *bflow* Toolbox* is restarted, there will be a new menu item from which the external program can be started. Once again, no knowledge about Eclipse programming is necessary.

Name	Install path	Parameter
SWI-Prolog	C:\Program Files (x86)\SWI-Prolog\bin\plcon.exe	
EpcTools	E:\epctools-auswertung\programm\epctools-norm...	
elementco...	E:\bflow\ElementCounter.jar	-f:$file

Fig. 4. User Interface for integrating third-party tools

Finally, we have to make sure that the results computed by the third-party program are transferred back into the *bflow* Toolbox* user interface. For this purpose, we provide several interfaces. They abstract away Eclipse implementation details and allow the external program

- to print information into the Eclipse console view,
- to add information about an error, warning or information to the Eclipse problem view
- to add a visual marker to a graphical model element,
- to add, delete or change attributes of the modeling elements

With the features described above, it is possible to integrate new functionality into the *bflow* Toolbox* without having to learn about Eclipse development. In many cases, new features can be added even without having to compile the *bflow* Toolbox* sources. We hope that these interfaces attract developers who are interested in adding functionality to the *bflow* Toolbox*.

5 Conclusions and Future Development

In this paper, we have presented some useful properties of our tool. Continuous verification and the table-based input wizards can be helpful for a modeler. Advanced users and programmers can take advantage of the possibilities to add functionality to the tool.

We are aware of the fact that future releases of the *bflow* Toolbox* will have to contain more and improved functionality, in particular in the area of model management (finding, comparing and integrating models from large model repositories). The *bflow* Toolbox* sources and executables for Windows and Linux can be downloaded from `www.bflow.org`. Researchers and practitioners are invited to download, use and improve the tool.

References

1. Wynn, M.T., Verbeek, E., van der Aalst, W.M.P., Edmond, D.: Business process verification - finally a reality! Business Process Management Journal **15** (2009) 74–92
2. Fahland, D., Favre, C., Jobstmann, B., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Instantaneous soundness checking of industrial business process models. In: Business Process Management, 7th International Conference, BPM 2009. Volume 5701 of LNCS., Springer (2009) 278–293
3. Gruhn, V., Laue, R.: A heuristic method for detecting problems in business process models. Business Process Management Journal **16** (2010)
4. Laue, R., Kühne, S., Gadatsch, A.: Evaluating the Effect of Feedback on Syntactic Errors for Novice Modellers. In: EPK 2009, Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten. CEUR Workshop Proceedings (2009)
5. Gruhn, V., Laue, R., Kühne, S., Kern, H.: A business process modelling tool with continuous validation support. Enterprise Modelling and Information Systems Architecture **4** (2009)

6. Favre, C., Gschwind, T., Koehler, J., Kleinöder, W., Maystrenko, A., Muhidini, K., Völzer, H., Wong, J.: Faster and better business process modeling with the IBM pattern-based process model accelerators. In: Business Process Management Demonstration Track 2009. (2009) 483–498
7. Mazanek, S., Minas, M.: Business process models as a showcase for syntax-based assistance in diagram editors. In: MoDELS. Volume 5795 of LNCS., Springer (2009) 322–336
8. Kern, H., Kühne, S.: Model interchange between ARIS and Eclipse EMF. In Tolvanen, J.P., Gray, J., Rossi, M., Sprinkle, J., eds.: 7th OOPSLA Workshop on Domain-Specific Modeling at OOPSLA 2007. (2007)
9. Kern, H., Kühne, S.: Integration of Microsoft Visio and Eclipse Modeling Framework using M3-level-based bridges. In: 2nd ECMDA Workshop on Model-Driven Tool & Process Integration, Enschede, Netherlands (2009)

Scenario-based process modeling with GRETA

Dirk Fahland[1] and Matthias Weidlich[2]

[1] Humboldt-Universität zu Berlin, Germany
`fahland@informatik.hu-berlin.de`
[2] Hasso-Plattner-Institute, University of Potsdam, Germany
`matthias.weidlich@hpi.uni-potsdam.de`

Abstract. Designing understandable business process models is one of the key factors to successful business process management. Current modeling practices advocate the use of block-oriented concepts and subprocesses to structure complex process models. However, such guidelines cannot be applied in any case as case studies in process mining have shown. Previously, we proposed the *scenario-based* paradigm to structure models of complex processes in behavioral fragments, i.e., scenarios. This paper presents GRETA as a tool that supports scenario-based process modeling and execution.

1 Introduction

Business process modeling has become an established technique for documenting, understanding, and analyzing business processes of enterprises of different kinds and size. Depending on the domain, process modeling results in several hundred or thousand process models [1]. These models need to be created, understood, and maintained by modelers and stakeholders who are not necessarily experts in computer science.

The BPM community developed modeling practices to create understandable models [2]. Typically large process models are structured hierarchically using subprocess of reasonable size or block-oriented concepts. However, important lessons from process mining tell that actual processes do not follow such structuring. Typically, flow of information connects various parts of a business process leading to "spaghetti-like" process models [3] for which structuring into subprocesses or block-oriented concepts is hard, if possible at all, e.g., Fig. 3 on the right.

In [4], we proposed the *scenario-based* modeling paradigm to structure behavior of processes into *behavioral fragments* of any chosen shape. In this paradigm, a process model is a set of *scenarios*. Each scenario denotes a finite acyclic process fragment consisting of several actions, some of them ordered in parallel. A distinguished *history* of the scenario expresses when the scenario may occur, i.e., how to *continue a process instance* that exhibited the scenario's history. We formalized these notions in the model of *oclets* [5] based on Petri nets. Figure 1 depicts two example oclets *depart standard* and *depart devices* (ignore the highlighted nodes). Each oclets's history is drawn above the horizontal line. Thus, *depart standard* triggers *depart devices* because of *present equipment*. Moreover, *depart standard* and *depart devices* describe different continuations after *present equipment*, i.e., both oclets describe *alternative continuations* of a
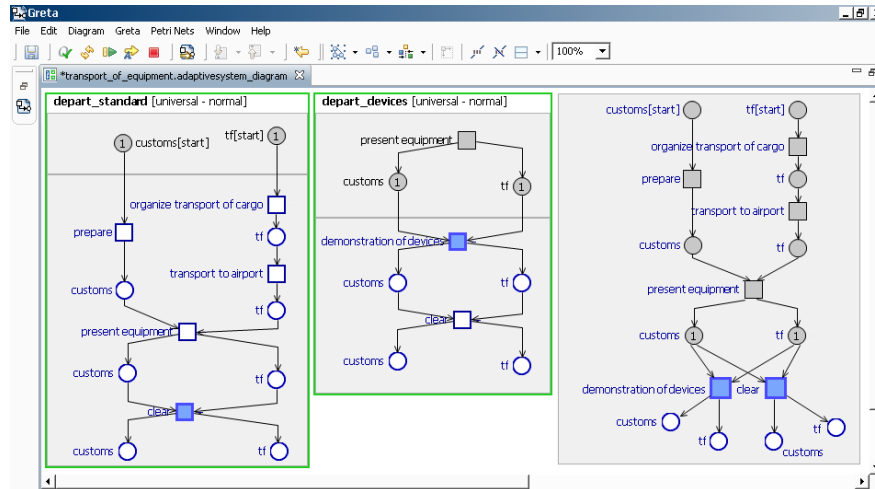
Fig. 1. Screenshot of GRETA with two oclets and a process instance constructed from these oclets.

process instance. This interplay allows to model complex business processes in terms of behavioral scenarios.

A process modeler reads each oclet as a "self-contained story" that occurs in the process. So, a scenario-based process model structures a complex process in terms of the "stories in the process" rather than blocks or subprocesses. Each story can, in principle, be understood in isolation. The difficulty of this approach is to design all oclets of the process so that they "fit together". This paper presents the tool GRETA that supports scenario-based process modeling in a *graphical editor* with animated *process execution*. Using GRETA, a process modeler can create and improve a complex model iteratively. Section 2 elaborates on the use case in more detail. Section 3 explains how GRETA supports scenario-based process modeling. We compare GRETA's key features to existing tools and conclude the paper in Section 4.

2 Use case: modeling unstructured processes

We illustrate the need for scenario-based modeling by a process that is run by the Task Force Earthquakes of the German Research Center for Geosciences (GFZ). The main purpose of the task force is to coordinate the allocation of an interdisciplinary scientific-technical expert team after catastrophic earthquakes worldwide [4]. In particular, we focus on the "Transport of Equipment" process of the task force. It specifies how the transport of scientific equipment from Germany to the disaster area is handled.

In order to elicit a process model for the "Transport of Equipment" process, typically, a process analyst conducts workshops with members of the task force and interviews them about the process. As a result, the analyst is faced with descriptions of how the transport of equipment is done in general. These descriptions resemble *stories* as they relate to the concrete experience of the team members gained during recent missions.

In our example, one story relates to preparation activities enacted in Germany, such as organizing transport of cargo and the actual transportation to the airport. Another story

relates to the activities that are done immediately after arrival in the disaster area, e.g., buying maps and renting vehicles. Once the standard processing has been clarified, the analyst discusses exceptional cases with the team members. Again, these descriptions can be seen as stories that describe 'what if' scenarios. In our exemplary process, for instance, customs might require that the equipment is presented before the decision on clearance is taken. Based thereon, the analyst abstracts the actual process model containing the complete standard processing along with all exceptional cases. In case of the "Transport of Equipment" process, the resulting model shows a complex structure as illustrated in Fig. 3 on the right.

3 Tool support for scenario-based modeling

How to model processes with GRETA**.** The scenario-based modeling approach differs from the typical approach discussed in Sect. 2 in the final step. Instead of creating one Petri net that copes with all eventualities, the modeler expresses each story of the "Transport of Equipment" process as a scenario. These scenarios are then assessed for continuations and dependencies. GRETA supports scenario-based process modeling with *oclets* by a graphical editor. To express a scenario in GRETA, the modeler creates a new oclet and models the scenario's behavior as a Petri net in the oclet's lower compartment as shown in Fig. 1 on the left and middle. To express when the scenario may occur, the modeler describes the *history* that triggers the scenario in the oclet's upper compartment. Each oclet is created *locally*, focusing on its story and its local history.

Typically, standard scenarios such as *depart standard* of Fig. 1 are modeled first. Then, variants and exceptions like *depart devices* are introduced one oclet at a time. The modeler may refine the oclets iteratively to achieve consistency of the overall processing. To this end, GRETA provides animated execution of oclets. The modeled oclets can be executed step-wise until a situation is reached in which a particular oclet should "fit", i.e., can be executed. If this is not the case, GRETA allows pausing the animation, changing the oclet's history, and resuming the animation at the current state. Thus, GRETA supports the modeler effectively in relating different scenarios to each other.

The tool GRETA**.** Having illustrated how GRETA supports scenario-based process modeling we now explain some technical aspects of the tool. We designed GRETA in a model-driven approach: data structures (oclets, process instances) and the graphical editor were modeled in respective languages of the *EMF/GMF framework* of *Eclipse* (www.eclipse.org). These models were translated to executable code which we extended manually to improve usability; details are given in [6].

Fig. 2. Architecture of GRETA.

GRETA exposes its data structures and user interface as *plugins* as illustrated in Fig. 2.

GRETA's *animated execution plugin* provides an *execution engine* for scenario-based models. When started, the engine (1) determines the *enabling condition* of each action in the denoted oclets. For instance, in Fig. 1 action *demonstration of devices* is enabled iff *present equipment* occurred. Then, the engine (2) checks which actions are enabled in the
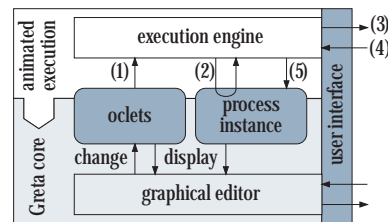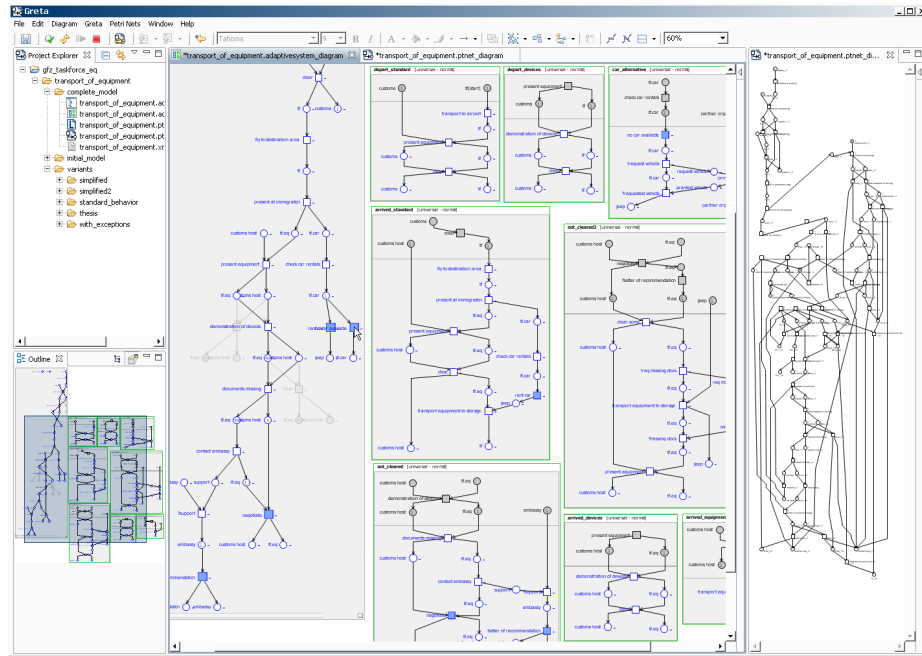
Fig. 3. The complete model of the "Transport of Equipment" process of the "Taskforce Earthquakes" (in the middle) and the Petri net model that can be synthesized with GRETA (on the right).

current process instance and (3) presents all enabled actions to the user by highlighting them. In Fig. 1, the current process instance is depicted on the right as gray shaded nodes with thin borders. Actions *demonstration of devices* and *clear* are enabled and highlighted as possible extensions of this process instance. The user (4) picks one enabled action for execution by clicking on it, and the engine (5) extends the process instance by the chosen action. The user may pause, stop, or reset the execution at any time to change the modeled oclets as she wishes.

Scalability. GRETA can handle models of complex processes consisting of dozens of scenarios with various exceptions. Fig. 3 shows a typical modeling situation in GRETA. The central window shows all oclets of the "Transport of Equipment" process of the "Taskforce Earthquakes" and a process instance on the left with four enabled actions. The two actions on the right are alternatives, all other actions are enabled concurrently. All enabled actions originate in different oclets as highlighted.

As an extension, GRETA supports *anti-oclets* which describe behavior that *must not* occur in the process. Further plugins for GRETA allow the modeler to *check consistency* of the model (i.e., whether each action of each oclet can actually be triggered by some other oclet), to *verify soundness* of a process model, and to automatically *synthesize a classical Petri net-based process model* such as the one depicted in Fig. 3 on the right. The synthesized Petri net exhibits exactly the behavior modeled in the oclets. Though the net is not necessarily "elegant" because the modeled real-world process has many exceptions and alternatives and does not follow a block structure. In Fig. 3, the oclets are structurally simpler and hence easier to comprehend than the Petri net.

55

4 Discussion and Conclusion

This paper presented GRETA as a *proof of concept prototype tool* that demonstrates the feasibility of *scenario-based process modeling and execution*. Currently, we are successfully applying GRETA in a case study where we model the process of the "Taskforce Earthquakes" with oclets as explained in Sect. 2 and 3. The case study itself is work in progress; some more information can be found in [4, 7]. GRETA and all its source code together with several examples are available for download at http://service-technology.org/greta.

Related tools and approaches. First and foremost, our work relates to approaches of scenario-based process modeling. Desel et al. advocated to model a business process in terms of its partially ordered runs [8] or expressions over finite scenarios [9]. The VipTool allows to synthesize a Petri net-based process model from such a scenario-based model [9]. Oclets extend these ideas and provide a mechanism to control the chaining of scenarios in terms of local histories. GRETA directly executes oclet-based models. Both ideas are influenced by *Live Sequence Charts* and the *play engine* [10]. In comparison to these, oclets directly serve business process modeling as they are based on Petri nets.

A closely related approach are *proclets* [11]. A proclet is a small workflow net in which actions send and receive data via channels. A process model is a set of proclets that are instantiated and coupled along their channels according to an underlying business object model. In [12], process behavior is described with a set of business object life cycle models. In both approaches, interaction between process artifacts follows from a data model, whereas *oclets* focus on control flow and use the notion of a *history* to describe when a behavior may occur; anti-oclets provide additional expressive power.

A high degree of variation of a common business process might be addressed using concepts of flexible process management. Research projects like ADEPT [13] and WASA [14] developed process management systems that enable ad-hoc modification of a process model for certain process instances. The YAWL workflow engine provides *worklets* [15], i.e., subprocesses that can be chosen and instantiated at runtime. The main differences between GRETA and existing execution engines root in the underlying semantic model: oclets are defined on the semantic model of *distributed runs* and the process instance's *history* defines which actions are enabled [5]. Existing execution engines operate on *sequential runs* and decide enabling of actions based only on the current *state*: history information is not available. The main difference to ADEPT is that ADEPT requires symmetrical, block-structured process models [13]. In contrast, oclets are unconstrained and describe process flow "as is". So, integrating oclets into an existing engine, e.g., as a plugin, remains a challenging and interesting task.

Finally, oclets can be seen as reusable patterns from which a process model is derived. Patterns have been proposed to business process modeling on various abstraction levels, e.g., control-flow patterns [16], business semantics aware action patterns [17], activity patterns that represent micro workflows [18]. Such patterns might be leveraged for modeling support, for instance, to accelerate process modeling and minimize modeling errors [19]. In general, oclets are more specific than the aforementioned patterns. They focus on partial scenarios and dynamically constructing process instances from scenarios whereas patterns aim at reuse or modeling support in a broader context.

Future work. In future work, we aim at lifting oclets to high-level modeling languages, such as BPMN. Here, the question of appropriate notions of a history has to be answered. Further on, the execution of scenarios involves the decision on a dedicated continuation once multiple oclets are activated. Such a decision should be supported by a tool suggesting the best continuation relative to a quality criterion (e.g., execution time).

References

1. Rosemann, M.: Potential pitfalls of process modeling: part A. Business Process Management Journal **12**(2) (2006) 249–254
2. Mendling, J., Reijers, H.A., van der Aalst, W.M.P.: Seven process modeling guidelines (7pmg). Information & Software Technology **52**(2) (2010) 127–136
3. van der Aalst, W.M.P., Rubin, V., Verbeek, H.M.W., van Dongen, B.F., Kindler, E., Günther, C.W.: Process mining: a two-step approach to balance between underfitting and overfitting. Software and System Modeling **9**(1) (2010) 87–111
4. Fahland, D., Woith, H.: Towards process models for disaster response. In: Workshops of the BPM'08. Volume 17 of LNBIP., Springer-Verlag (2008) 244–256
5. Fahland, D.: Oclets - scenario-based modeling with Petri nets. In: Petri Nets 2009. Volume 5606 of LNCS., Paris, France, Springer-Verlag (June 2009) 223–242
6. Wolf, M.: Erstellung einer modellbasierten Laufzeitumgebung für adaptive Prozesse. Diplomarbeit, Humboldt-Universität zu Berlin (September 2008)
7. Weidlich, M., Zugal, S., Pinggera, J., Fahland, D., Weber, B., Reijers, H., Mendling, J.: The impact of change task type on maintainability of process models. In: ER-POIS 2010, held in conjunction with CAiSE 2010. (2010) to appear.
8. Desel, J.: Validation of process models by construction of process nets. In van der Aalst, W.M.P., Desel, J., Oberweis, A., eds.: BPM. Volume 1806 of LNCS., Springer (2000) 110–128
9. Bergenthum, R., Desel, J., Mauser, S., Lorenz, R.: Construction of process models from example runs. T. Petri Nets and Other Models of Concurrency **2** (2009) 243–259
10. Harel, D., Marelly, R.: Come, Let's Play: Scenario-Based Programming Using LSCs and the Play-Engine. Springer-Verlag (2003)
11. van der Aalst, W.M.P., Barthelmess, P., Ellis, C.A., Wainer, J.: Workflow Modeling using Proclets. In: CoopIS 2000. Volume 1901 of LNCS., Springer (2000) 198–209
12. Küster, J.M., Ryndina, K., Gall, H.: Generation of business process models for object life cycle compliance. In: BPM 2007. Volume 4714 of LNCS., Springer (2007) 165–181
13. Dadam, P., Reichert, M.: The ADEPT project: a decade of research and development for robust and flexible process support. Computer Science - R&D **23**(2) (2009) 81–97
14. Vossen, G., Weske, M.: The WASA2 object-oriented workflow management system. In: SIGMOD Conference 1999, ACM Press (1999) 587–589
15. Adams, M., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Worklets: A service-oriented implementation of dynamic flexibility in workflows. In: OTM. Volume 4275 of LNCS., Springer (2006) 291–308
16. van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P.: Workflow patterns. Distributed and Parallel Databases **14**(1) (2003) 5–51
17. Smirnov, S., Weidlich, M., Mendling, J., Weske, M.: Action patterns in business process models. In: ICSOC. Volume 5900 of LNCS. Springer (2009) 115–129
18. Thom, L.H., Reichert, M., Iochpe, C.: Activity patterns in process-aware information systems: Basic concepts and empirical evidence. IJBPIM **4**(2) (2009) 93 – 110
19. Gschwind, T., Koehler, J., Wong, J.: Applying patterns during business process modeling. In: BPM. Volume 5240 of LNCS., Springer (2008) 4–19

The PrICE Tool Kit: Tool Support for Process Improvement

Mariska Netjes, Hajo A. Reijers, Wil M.P. van der Aalst

Eindhoven University of Technology,
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{m.netjes,h.a.reijers,w.m.p.v.d.aalst}@tue.nl

1 The PrICE Approach

Process improvement is an important means to obtain competitive advantage and improve customer satisfaction. The PrICE tool kit provides support for process improvement and has been developed to show the feasibility of the PrICE approach. The approach for Process Improvement by Creating and Evaluating process alternatives (in short: the PrICE approach) describes the concrete steps that have to be taken to get from the *as-is* process to the *to-be* process. A common view on process improvement roughly distinguishes four phases: (1) framing the process of interest, (2) understanding the current (*as-is*) process, (3) designing the new (*to-be*) process, and (4) implementing the new process [15]. Many approaches and methods for process improvement are used in practice, but most of these do not address the concrete design of an improved process. The PrICE approach supports phase (3) of a process improvement project: designing the *to-be* process [15].

The *as-is* model, the input of the approach, is the result of phase (2) of a process improvement project: understanding the *as-is* process [15]. The *as-is* model should contain information on the control flow, the data, the resources and the performance of the process. The PrICE approach consists of four steps:

1 **Find applicable redesign operations**: a redesign operation supports a particular type of redesign creation. Applicable operations can be found with process measures or process mining. Process measures provide a global view on the characteristics of the process and their values may reveal weaknesses in the process [5]. Process mining provides a powerful means to find bottlenecks and other redesign opportunities in the process [1, 2].
2 **Select suitable process parts**: specific parts of the process model that can be redesigned with one or more of the applicable redesign operations are identified. Process mining can also be used to support this step of the approach. In addition, requirements are set on the process parts that can be selected for redesign to be able to create correct alternative models. The user is guided in the selection of such process parts.
3 **Create alternative models**: the applicable redesign operations are performed on selected process parts, thus, creating alternative process models.

A formal foundation for the creation of process alternatives is developed to ensure the correctness and to provide a base for the implementation of the tool kit. This formal foundation has been published in [6].

4 **Evaluate performance of alternatives**: the created alternative models are simulated to predict their expected performance. By comparing the simulation results, a quantitatively supported choice for the best alternative model can be made. A simulation plan has been published in [4]. The developed tool support enables simulation in batch, i.e., the simulation of any number of alternatives without user interaction.

The output of the approach is a model of the *to-be* process which is selected from the alternative models based on the performance evaluation. This *to-be* process is the input for phase (4) of a process improvement project: implementing the new process [15].

2 Relevance

Current redesign practice is performed in a highly participative fashion where management consultants encourage business professionals within a workshop setting to think of one or more alternatives for the *as-is* process. The role of the external consultants is to moderate the workshop, to stimulate people to abandon the traditional beliefs they may have about the process in question and to mobilize support for the upcoming changes. Sharp and McDermott, for instance, describe the lack of methodological support for this practice as follows: "How to get from the *as-is* to the *to-be* [in a BPR project] isn't explained, so we conclude that during the break, the famous ATAMO procedure is invoked – And Then, A Miracle Occurs" [15]. The consequence of this lack of support is that the design of the *to-be* process becomes a subjective and non-repeatable act resulting in abstract process designs without an accurate estimate of the expected gains. Because the same steps are followed in our approach that are present in the current practice of process redesign, it seems viable to support the interaction between business professionals with the PrICE approach and tool kit. The application scenario we envision is that in a workshop a set of attractive redesign alternatives is created with support of the PrICE tool kit. The creation of alternative models is a highly interactive activity. A process model can never capture all information that is relevant for process redesign. The user is involved to ensure that the alternative models are feasible. The tool automates the parts that do not need user interaction and supports the user in creating alternative models in a systematic manner. During a break or afterwards, all or a selection of these alternative designs are simulated in batch, i.e., without further user interaction.

Business Process Management (BPM) systems provide a broad range of facilities to enact and manage operational business processes. Ideally, these systems should provide support for the complete BPM life-cycle: (re)design, configuration, execution, control, and diagnosis of processes. However, based on an extensive evaluation of the FileNet P8 BPM Suite, we have show in [1] that existing

BPM tools are unable to support the full life-cycle. Especially the diagnosis and the (re)design phases are not sufficiently supported. Diagnostic support is lacking for the search for weaknesses in the process and the generation of improvement suggestions. Furthermore, in the design phase, the creation of the redesign alternatives is not supported. The PrICE approach provides an integrated approach for the diagnosis and the design of business processes. The first two steps of the PrICE approach provide support for the diagnosis phase while all steps support the redesign part of the design phase [1].

3 Main Features

The main features of the PrICE tool kit are 1) the use of process mining to find redesign opportunities, 2) the user guidance in the selection of process parts, 3) the creation of process alternatives, 4) the construction of the process alternatives tree, and 5) the evaluation of the alternatives with simulation. Figure 1 depicts a screenshot of the user interface of the tool kit. The lower part displays
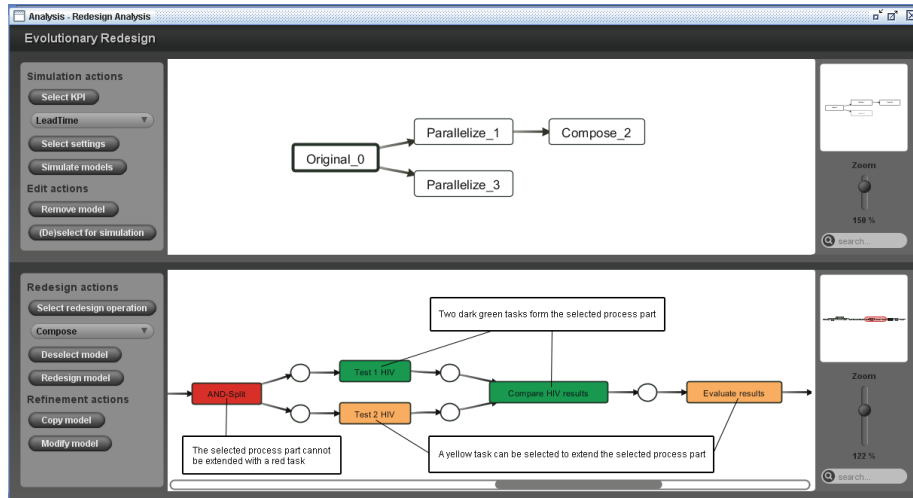

Fig. 1. A process part that is selected for the application of the compose operation

the process model for which an alternative will be created and provides the options to create a process alternative. After the selection of a redesign operation, a process part for redesign is selected by the user by clicking on the tasks in the process model. Colors are used to guide the user and show which tasks may be added to the current selection to form a process part (see Figure 1 for an illustration). This way, it is ensured that the input for the creation of a process alternative is such that a correct alternative model can be created. The upper part of Figure 1 shows the process alternatives tree. The selected node in the tree

corresponds to the model that is displayed in the lower part. After the creation of an alternative model, the tree is updated with a new node representing this alternative. The upper part of the user interface also provides the options for the evaluation of the alternatives in the tree. One can select a subset of nodes for simulation or simulate the complete tree. A simulation study is performed in batch, i.e., all selected models are simulated without user interaction. Afterwards, the simulation results are displayed on the tree nodes. In addition, colors are used to guide the user in finding the best performing alternative(s).

4 Architecture

The PrICE tool kit is implemented as part of the Process Mining (ProM) framework [9]. In ProM, a generic process format, called high-level (HL) model, is available to specify the control flow, data, resource and performance perspectives. Several modeling languages can be used to model such a HL model. Protos [7] is one of these languages. We implemented a HL model for a Protos model, which is called a HL Protos model. The use of one of the many mining plugins is another possible means to obtain a process model and process information. The discovery of a complete simulation model from an event log [13] is an example of this. Such a simulation model is implemented as a HL PetriNet model. We use Figure 2 to sketch the technical infrastructure of the tool kit. At the top left
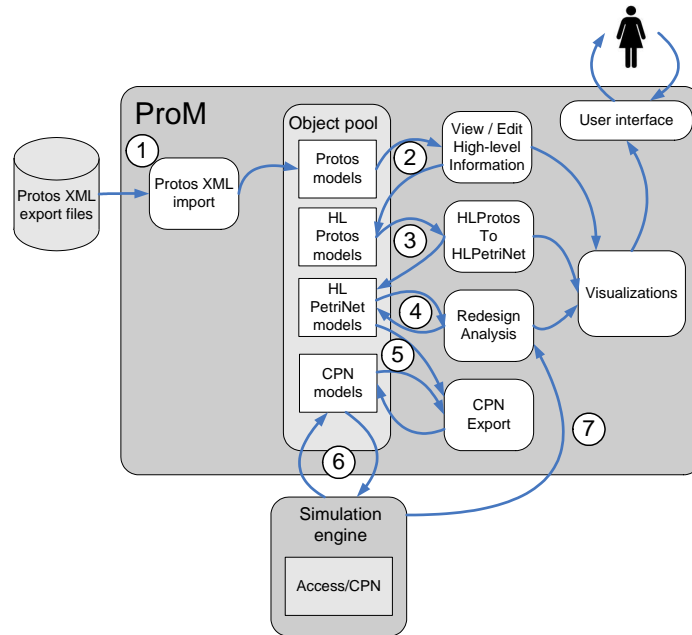

Fig. 2. Tool kit architecture

side of Figure 2, indicated with (1), a Protos model is imported to the object pool in ProM and converted to the HL format (see (2) in Figure 2). HL models are displayed by the Edit / View High-level Information plugin [13]. Then, the control flow of the HL Protos model is converted to a Petri net, thus creating a HL PetriNet model (see (3) in Figure 2). The functionality for the creation and evaluation of alternative models is implemented with the Redesign Analysis plugin (see (4) in Figure 2). For the evaluation of the performance of alternative models we use Colored Petri nets (CPNs). The collection of alternative models is converted to CPN models with the CPN Export [12] and analyzed using CPN tools [3] (see (5) in Figure 2). CPN Tools provides support for the modeling and simulation of business processes. A simulation engine for the automatic simulation of multiple CPN models is used for performance evaluation (see (6) in Figure 2). We built the simulation engine on the Access/CPN framework [16]. The simulation results are returned to the Redesign Analysis plugin (see (7) in Figure 2).

The PrICE tool kit has been developed as a research prototype. It has been implemented on top of the ProM framework [9] which supports among others process mining techniques [2], the storage and reuse of objects and the conversion of models. Furthermore, it is open source, making it easy to plug in new pieces of functionality. This allowed us to implement a rather mature prototype in terms of interoperability between the PrICE tool kit and other tools, user interface and supported modeling languages. The tool is freely available for download since mid 2009, but we are not aware of any use of it by others to support process improvement. So far, we tested the PrICE tool kit with a number of processes. We also tested whether it is feasible to create realistic redesign alternatives with the PrICE tool kit. For this test, we used a redesign project that is described in [11]. The project describes a real life business process that is executed at a mental healthcare institute and the creation of seven possible alternatives for the original process. We succeeded in reproducing five of these alternatives. This outcome gives a first indication that the PrICE tool kit is useful in supporting process redesign projects in practice. Currently, we are working together with Pallas Athena [7] to evaluate the approach and tool kit in real business settings.

5 Links

The PrICE tool kit is made available through download and through the SHARE system [14]. A download of the tool kit can be performed from [8] which provides the latest internal version of ProM 5. Additional information can be found on [10]. With the SHARE system, an environment to test and play with the tool kit is provided. The environment includes the tool, a tutorial, a screencast and several input models for the tool. The environment can be accessed from [10] after registration.

Acknowledgement

References

1. W.M.P. van der Aalst, M. Netjes, and H.A. Reijers. Chapter 4: Supporting the Full BPM Life-Cycle Using Process Mining and Intelligent Redesign. In K. Siau, editor, *Contemporary Issues in Database Design and Information Systems Development*, pages 100–132. IGI Global, Hershey, USA, 2007.
2. W.M.P. van der Aalst, H.A. Reijers, A.J.M.M. Weijters, B.F. van Dongen, A.K. Alves de Medeiros, M. Song, and H.M.W. Verbeek. Business Process Mining: An Industrial Application. *Information Systems*, 32(1):713–732, 2007.
3. CPN Tools website. http://wiki.daimi.au.dk/cpntools/cpntools.wiki.
4. M.H. Jansen-Vullers, P.A.M. Kleingeld, and M. Netjes. Quantifying the Performance of Workflows. *Information Systems Management journal*, 25(4):332–343, 2008.
5. M. Netjes, S. Limam Mansar, H.A. Reijers, and W.M.P. van der Aalst. Performing Business Process Redesign with Best Practices: An Evolutionary Approach. In J. Filipe, J. Cordeiro, and J. Cardoso, editors, *Enterprise Information Systems (9th International Conference, ICEIS 2007, Funchal, Madeira, June 12-16, 2007, Revised Selected Papers)*, volume 12 of *Lecture Notes in Business Information Processing*, pages 199–211. Springer-Verlag, Berlin, 2009.
6. M. Netjes, H.A. Reijers, and W.M.P. van der Aalst. On the Formal Generation of Process Redesigns. In D. Ardagna, M. Mecella, and J. Yang, editors, *Business Process Management Workshops: BPM 2008*, volume 17 of *Lecture Notes in Business Information Processing*, pages 217–228. Springer-Verlag, Berlin, 2009.
7. Pallas Athena website. www.pallas-athena.com.
8. ProM download website. http://prom.win.tue.nl/tools/prom/nightly5/.
9. ProM website. www.processmining.org.
10. Redesign in ProM website. http://prom.win.tue.nl/research/wiki/online/redesign.
11. H.A. Reijers. *Design and Control of Workflow Processes: Business Process Management for the Service Industry*, volume 2617 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2003.
12. A. Rozinat, R.S. Mans, M. Song, and W.M.P. van der Aalst. Discovering Colored Petri Nets from Event Logs. *International Journal on Software Tools for Technology Transfer*, 10(1):57–74, 2008.
13. A. Rozinat, R.S. Mans, M. Song, and W.M.P. van der Aalst. Discovering simulation models. *Information Systems*, 34(3):305–327, 2009.
14. SHARE website. http://fmt.cs.utwente.nl/redmine/wiki/5/SHARE.
15. A. Sharp and P. McDermott. *Workflow Modeling: Tools for Process Improvement and Application Development*. Artech House Publishers, Boston, 2009.
16. M. Westergaard and L.M. Kristensen. The Access/CPN Framework: A Tool for Interacting with the CPN Tools Simulator. In G. Franceschinis and K. Wolf, editors, *Applications and Theory of Petri Nets 2009*, volume 5606 of *Lecture Notes in Computer Science*, pages 313–322. Springer-Verlag, Berlin, 2009.

service-technology.org/live

Replaying tool experiments in a Web browser

Niels Lohmann

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

Abstract. In many fields of science, the quality and applicability of presented approaches heavily relies on experimental results. At the same time, such experiments are only described briefly and are typically boiled down to a single diagram or table. To maintain high scientific standards, experiments need to be transparent and repeatable. This not only provides a deeper understanding in the used tools, but also facilitates the improvement and comparison of approaches.

This paper introduces service-technology.org/live, a Web site which allows to replay experiments in a Web browser. This gives colleagues and reviewers the possibility to get detailed information on the experimental setup without the need of actually installing tools. The Web site further offers the used tools and the input data for download together with information on how to repeat the experiments locally.

1 Motivation

The formal verification of business process models forms a constant stream of research in the BPM community. With a variety of correctness criteria of which soundness is the most popular one, also a multitude of automatic verification tools were developed over the last years. The algorithms implemented by these tools typically have a devastating worst-case complexity (usually an exponential explosion in the size of the model) which seems to make any application beyond toy examples unreasonable. As a matter of fact, experiences in the field of model checking [1] reveal that even theoretically unfeasible algorithms may perform surprisingly well on real-world instances. Consequently, prototype implementations need to be validated with the help of case studies or experiments.

To follow good scientific standards, such experiments should not only be thoroughly *described*. To achieve maximal transparency and to simplify comparison and improvements, they also need to be *repeatable*. That said, not only the respective tools and input data need to be published, but also information on the installation and invocation are required to replay experiments. Unfortunately, only few papers provide such details and experimental results are typically boiled down to a few tables or diagrams, making the entire experiment and hence the implemented algorithm intransparent and vulnerable.

We identified several problems that hinder adequate transparency:

- *Page limits.* Technical details on the installation or invocation of tools can hardly be provided within the page limit of a typical research paper. At the same time, prototypes are often a byproduct of the respective research paper and hence dedicated tool papers are not yet available.
- *Availability of tools.* Many tools are proof-of-concept implementations for the respective research paper and are often not publicly available. The same holds for beta versions or commercial tools.
- *Installation of tools.* Scientific software development often lacks a professional release and documentation process which makes the installation complex and cumbersome.
- *Description of the experimental setup.* An experiment hardly involves a single call of a tool or "click" of a button. Often, only the interplay between several tools or parameters realize the experiment.
- *Availability of input data.* Even if all tools are available and the experimental setup is properly described, the concrete input data are hardly available for download. This is typically because of nondisclosure agreements or other legal constraints on industrial process models. But even if input data are public (e.g., published reference models), they are frequently converted beforehand or otherwise preprocessed.

This paper introduces service-technology.org/live, a Web site dedicated to the technical aspects of the described problems. This Web site not only "outsources" technical descriptions on the experimental setup, but also facilitates the replay of the experiments within a Web browser. The remainder of the paper is organized as follows. The next section gives an overview of service-technology.org/live. Section 3 discusses technical details. Finally, Sect. 4 gives directions of future work and concludes the paper.

2 Overview

The Web site service-technology.org/live hosts with BPEL2oWFN [6], Fiona [10], Wendy [7], Rachel [5], Marlene [3], and Mia [4] several of the tools developed by the theory of programming research groups at the University of Rostock and the Humboldt-Universität zu Berlin. Their focus is the correctness of business process and service models. We provide a detailed overview in [9]. As the tools are command-line based, their usage is — compared to tools with a graphical user interface — rather complicated. Furthermore, the tools follow the "one purpose, one tool" philosophy and can be combined in many ways to perform higher level tasks. To still be able to reproduce experimental results, we fixed several tool setups and offer them at service-technology.org/live, see Fig. 1.

In the simplest case, the user can choose an experiment which is described in the respective paper (i.e., choosing the name of a model from a drop-down list) and let the experiment be run "live" on the Web server. For other experiments, also a few parameters can be set. The output (see Fig. 2 for an example) consist of the following elements:
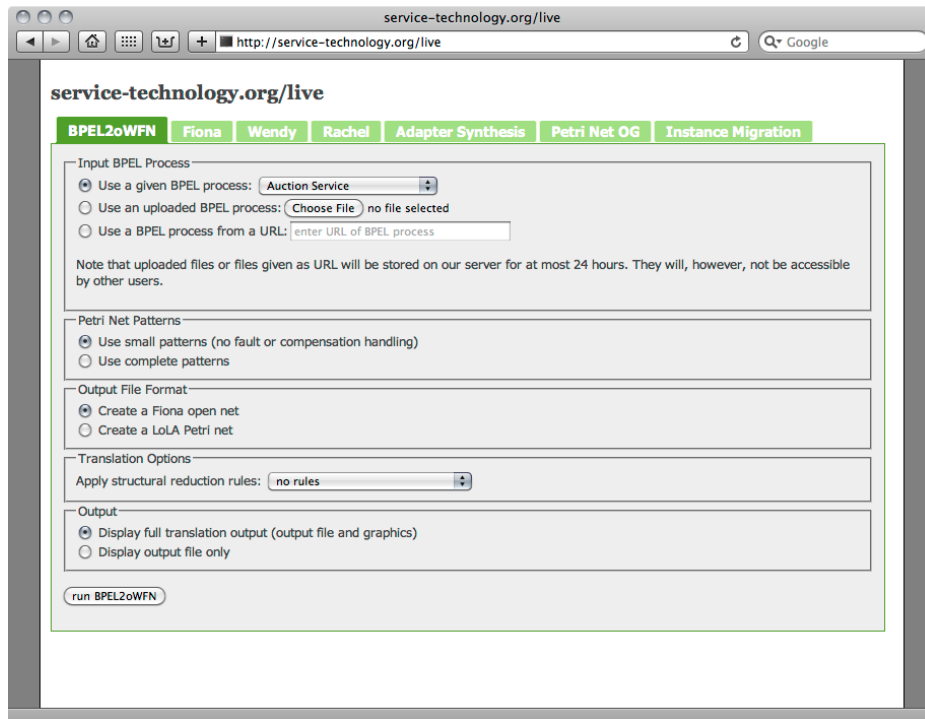
Fig. 1. Start page of service-technology.org/live.

- a description of the input of the experiment (e.g., the origin of the model),
- console windows (i.e., the output of the command-line tools),
- generated graphical representations (e.g., Petri nets or automata),
- a summary of invoked tools with their parameters, and
- download information for the involved models and tools.

Admittedly, the current state of interactivity is limited. We believe, however, that even this simplistic Web site already provides a decent amount of transparency for the experimental results. With a hosted experiment, we give colleagues and reviewers not only the opportunity to check for the *existence* of the tools, but also give an idea *how* the results are generated and also show the "look and feel" of the tools without the need of downloading or installing. This is, however, still possible, because all tools and input models are available online.

3 Technical realization

The Web site service-technology.org/live is written in PHP, but any other language that allows the Web server to invoke locally installed tools could be used instead.
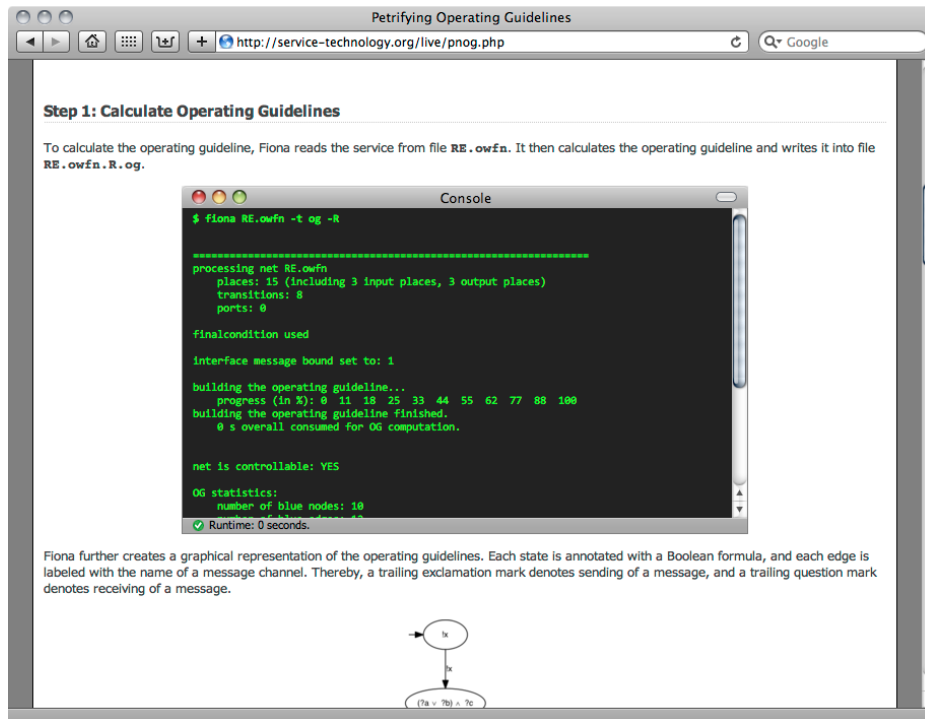
Fig. 2. Example for a console output.

The current server is an Apple Mac mini (1.83 GHz, 1 GB RAM) running an Apache 2.2 server with PHP 5.3.

The main components such as console windows or graphics generation and presentation have been encapsulated to facilitate extensions and the integration of new tools and experiments. The free JavaScript library jQuery[1] further realizes basic UI concepts such as tabbed windows and asynchronous loading of images using AJAX. The code of the Web site is free and open source.[2]

New experiments can be quickly integrated by performing two tasks:

1. Add a form to the front page to choose the input model and parameters.
2. Create a PHP script that runs and presents the experimental results.

Both steps are highly generic, and because of the encapsulation of server and language-specific details, we claim that anyone who is able to execute the tools in a console is likewise able to create such files for new experiments — assuming basic knowledge on HTML and PHP.

In the current state, no user access controls are implemented. All experiments are open for everybody. It is, however, possible to upload input models to the

---

[1] http://jquery.com/
[2] http://svn.gna.org/svn/service-tech/trunk/meta/live/

server or to provide a URL to a model. Such uploaded models are processed just like the hosted models, but are not shared with other users, and are deleted from the server regularly.

4   Conclusion

Concrete experiments hosted by service-technology.org/live have already been referenced in several published research papers [5,8,7]. We feel committed to use service-technology.org/live as a part of any future research paper which involves experimental results. By encapsulating repeatedly used functionality such as console windows or graphics generation, new results can be added without much programming effort. By publishing the source code of service-technology.org/live as open source, we hope that other research groups follow our example.

In future work, we plan to continue to simplify the addition of new experiments and tools. We also work on decoupling the hosted tools from the Web site and to offer them as *software as a service* using Web service standards such as SOAP and WSDL. Thereby, we can wrap complex verification tasks and allow them to be invoked remotely over the Web from other tools. We currently evaluate the use of the Petri net model checker LoLA [11] as soundness checker for the modeling platform Oryx [2].

References

1. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press (1999)
2. 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)
3. Gierds, C., Mooij, A.J., Wolf, K.: Reducing adapter synthesis to controller synthesis. IEEE T. Services Computing (2010), (accepted for publication in July 2010)
4. Liske, N., Lohmann, N., Stahl, C., Wolf, K.: Another approach to service instance migration. In: ICSOC 2009. pp. 607–621. LNCS 5900, Springer (2009)
5. Lohmann, N.: Correcting deadlocking service choreographies using a simulation-based graph edit distance. In: BPM 2008. pp. 132–147. LNCS 5240, Springer (2008)
6. Lohmann, N.: A feature-complete Petri net semantics for WS-BPEL 2.0. In: WS-FM 2007. pp. 77–91. LNCS 4937, Springer (2008)
7. Lohmann, N., Weinberg, D.: Wendy: A tool to synthesize partners for services. In: PETRI NETS 2010. pp. 297–307. LNCS 6128, Springer (2010)
8. Lohmann, N., Wolf, K.: Petrifying operating guidelines for services. In: ACSD 2009. pp. 80–88. IEEE Computer Society (2009)
9. Lohmann, N., Wolf, K.: How to implement a theory of correctness in the area of business processes and services. In: BPM 2010. LNCS, Springer (2010), (in press)
10. Massuthe, P., Weinberg, D.: Fiona: A tool to analyze interacting open nets. In: AWPN 2008. pp. 99–104. CEUR Workshop Proceedings Vol. 380, CEUR-WS.org (2008)
11. Wolf, K.: Generating Petri net state spaces. In: ICATPN 2007. pp. 29–42. LNCS 4546, Springer (2007)