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].

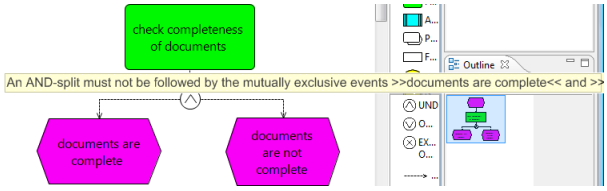

Fig. 1. Feedback on errors in the labels of the events

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*.

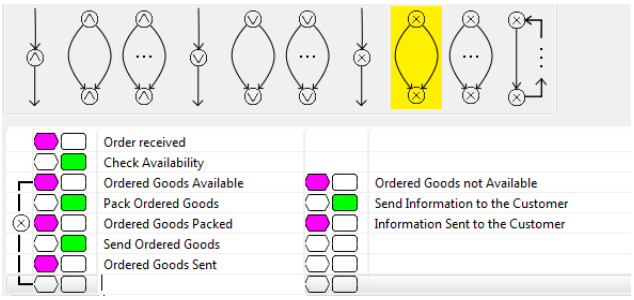

Fig. 2. User Interface for creating parts of the model very quickly

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)