# 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:

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

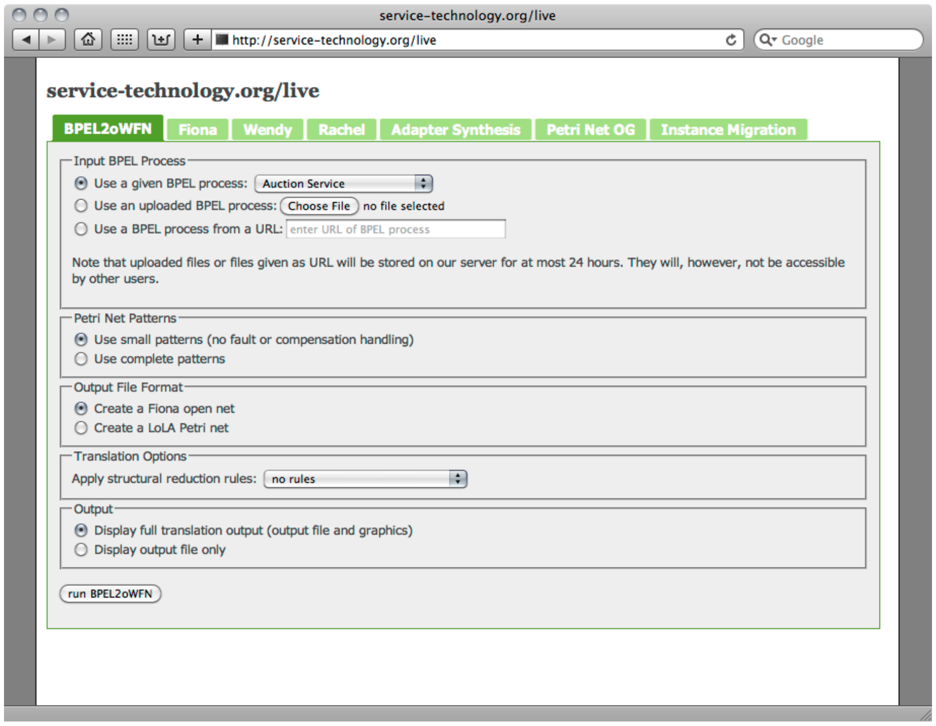
– *Page limits.* Technical details on the installation or invocation of tools can hardly be provided within the page limit of a typical research paper. At the same time, prototypes are often a byproduct of the respective research paper and hence dedicated tool papers are not yet available.
– *Availability 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:


service-technology.org/live
http://service-technology.org/live
Google

service-technology.org/live

BPEL2oWFN | Fiona | Wendy | Rachel | Adapter Synthesis | Petri Net OG | Instance Migration

Input BPEL Process
[✓ Use a given BPEL process:] Auction Service
Use an uploaded BPEL process: Choose File no file selected
Use a BPEL process from a URL: enter URL of BPEL process

Note that uploaded files or files given as URL will be stored on our server for at most 24 hours. They will, however, not be accessible by other users.

Petri Net Patterns
[✓ Use small patterns (no fault or compensation handling)]
Use complete patterns

Output File Format
[✓ Create a Fiona open net]
Create a LoLA Petri net

Translation Options
Apply structural reduction rules: no rules

Output
[✓ Display full translation output (output file and graphics)]
Display output file only

run BPEL2oWFN

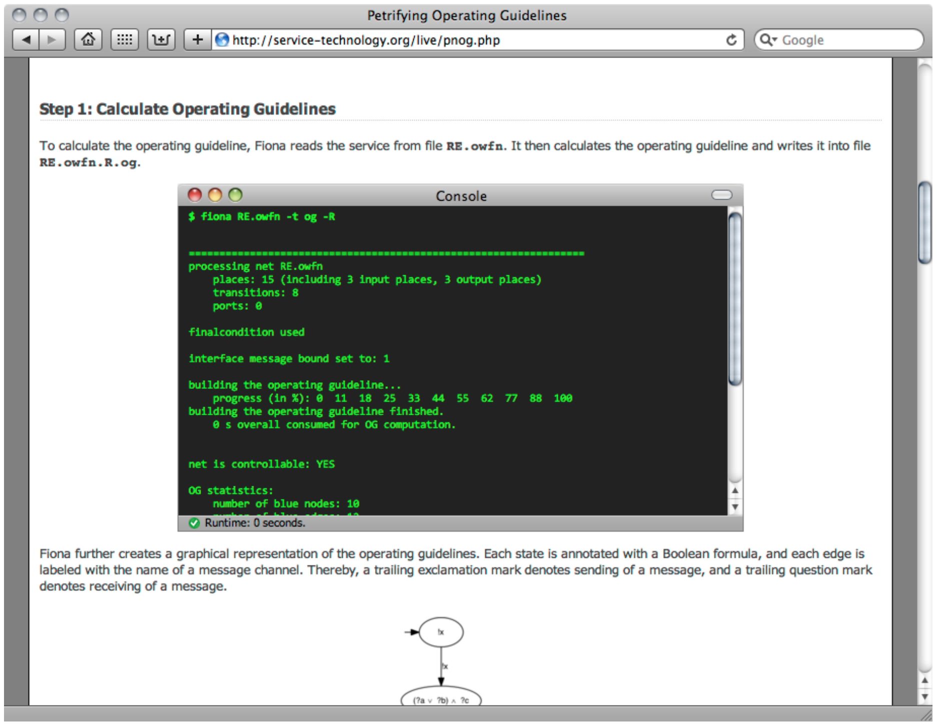
**Fig. 1.** Start page of service-technology.org/live.

– 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.


Petrifying Operating Guidelines
http://service-technology.org/live/pnog.php
Google
Console
!x
!x
(?a ∨ ?b) ∧ ?c

Step 1: Calculate Operating Guidelines

To calculate the operating guideline, Fiona reads the service from file **RE.owfn**. It then calculates the operating guideline and writes it into file **RE.owfn.R.og**.

```
$ fiona RE.owfn -t og -R


=================================================================
processing net RE.owfn
    places: 15 (including 3 input places, 3 output places)
    transitions: 8
    ports: 0

finalcondition used

interface message bound set to: 1

building the operating guideline...
    progress (in %): 0  11  18  25  33  44  55  62  77  88  100
building the operating guideline finished.
    0 s overall consumed for OG computation.


net is controllable: YES

OG statistics:
    number of blue nodes: 10
```

Runtime: 0 seconds.

Fiona further creates a graphical representation of the operating guidelines. Each state is annotated with a Boolean formula, and each edge is labeled with the name of a message channel. Thereby, a trailing exclamation mark denotes sending of a message, and a trailing question mark denotes receiving of a message.

**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)