KU LEUVEN

FACULTY OF ECONOMICS AND BUSINESS

Straightforward Petri net-based event log generation in ProM

vanden Broucke S, Vanthienen J, Baesens B.

KBI_1417

Straightforward Petri Net-Based Event Log Generation in ProM

Seppe K.L.M. vanden Broucke, Jan Vanthienen and Bart Baesens

Department of Decision Sciences and Information Management, KU Leuven
Naamsestraat 69, B-3000 Leuven, Belgium
seppe.vandenbroucke@kuleuven.be

TECHNICAL REPORT In this report, we present a ProM plugin which allows for straightforward event log generation, using token-based simulation driven by Petri net models. Although a large number of tools already exist for the simulation and analysis of Petri nets (CPN Tools being among the most notable), no technique exists which allows for the rapid generation of event logs (a collection of execution traces) based on a user-supplied Petri net in a straightforward manner, which would be a helpful addition within the research area of process mining, as this research field deals with the extraction of knowledge from such event logs, which are regarded as the focal point of analysis. Our proposed tool focuses on ease of use, provides different simulation strategies and configuration options covering most standard use cases, and outputs the desired event log in a format which can immediately be utilized in subsequent steps within a process mining analysis workflow.

Key words: process mining, event log generation, petri net simulation, ProM

1 Introduction

Process mining encompasses the research field which deals with the extraction of knowledge from event logs as recorded by (process aware) information systems [1]. Within the process mining area, a distinction is made between three broad, prominent tasks. First of all, process discovery deals with the automated construction of a process model out of an event log (i.e. there is no a-priori model). Second, conformance checking starts from a process model and an event log (there is an a-priori model) and analyzes the quality of the process model based in terms of its ability to represent the behavior in the event log, or attempts to explain where deviations from a designed or a prescribed model exactly occur and why. Third and finally, process enhancement also assumes the presence of an existing process model (perhaps discovered in an earlier step), but tries to improve or extend this model based on additional or more recent data.

Although event logs are regarded to be the focal point of analysis within the field of process mining, no straightforward approach or tool exists to construct these data repositories in a synthetic manner. However, such technique could be

useful in many academic and other use cases. To generate synthetic event logs, scholars and practitioners have, so far, mainly resorted to using the following tools and approaches. Firstly, CPN Tools [2], which can be regarded as, by far, the most complete and advanced suite for coloured Petri net modeling, simulation and analysis, but comes with a steep learning curve and is hard to use in the use case of modeling a (non-coulored) Petri net and deriving a collection of simulated traces. Second, the Process Log Generator [3], which—despite the name—randomly generates *models* based on user-supplied criteria and can also provide a related simulated event log, but does not provide means to modify the generated model or change simulation options. The final approach consists of direct construction of event logs without any formal semantic model driving a simulation, other than perhaps some stochastic statistical process (e.g. randomly constructing traces from a pool of activity labels). This last approach has the drawback that the constructed synthetic event logs are of less use, as process mining techniques assume that there is some underlying process model driving the execution of activities.

To help remedy this gap within the current offering of tools, we present a ProM plugin which allows for straightforward event log generation using token-based simulation driven by Petri net models. ProM is an academic, open-source framework for process mining and is used by researchers and academics world-wide [4, 5]. Our proposed tool focuses on ease of use, provides different simulation strategies and configuration options covering most standard use cases, and outputs the generated event logs in a native and standardized XES format[1], which can be immediately utilized in subsequent steps within a process mining analysis workflow.

The remainder of this technical report is structured as follows. Section 2 provides an encompassing overview of our proposed plugin, consisting of a description of objectives, functionality, architecture, and use cases. This section also provides a comparative overview with other tools and techniques. Finally, we provide installation instructions describing how to retrieve and install the tool. Our generation plugin is provided as free and open-source software. Section 3 concludes the report.

Note that we assume readers to be familiar with the field of process mining and its well-known concepts. More specifically, we expect that the concepts of event logs [1] and Petri nets [6] are known, and that the reader has some experience with the use of ProM.

2 Tool Overview

Following subsections addresses the following issues: first, the objectives of our presented ProM plugin are given, followed by an overview of the tool's functionality. Next, we describe the architecture of the developed plugin, before presenting

[1] Extensible Event Stream, see: `http://www.xes-standard.org/xesstandarddefinition`.

some use cases illustrating the working an use of the plugin. Following hereafter, we perform a comparison between our plugin and other related techniques and tools, before closing with installation instructions.

2.1 Objectives

This subsection outlines a description of the purpose and applicative domain of the tool.

In a nutshell, the main objective of our developed tool was to offer an easy way to generate an event log from a given Petri net model which also enables user to configure some general options—relevant to process mining practitioners and researchers—and outputs an event log which is directly useable in ProM without having to perform any intermediary steps. Concerning the supplied Petri net, we have deliberately chosen to regard the modeling of such models as being out of scope for our plugin, as there exist many excellent tools already which allow for the rapid modeling of such models, which can easily be imported in ProM. However, the simulation of an event log in these tools is seldom supported in a user-friendly manner.

The development of this plugin has followed from a real "need" experienced by the authors when setting up experiments to compare the performance of process mining discovery and other algorithms, which—in most scenarios—are composed out of both real-life and synthetic event logs. We thus can enumerate our objectives as follows:

Integration in ProM The plugin should be integrated into ProM, rather than a stand-alone package, as the ProM framework is heavily utilized by process mining researchers and practitioners;

Relevant configuration options The plugin does not allow to modify supplied Petri net models, as there exists many tools already supporting the creation (and validation) of such models. However, the plugin should allow to configure relevant options, such as the visibility of transitions;

Native event log format Generated event logs should be created as XES files, the event log format native to ProM. Many tools already allow to generate event logs as key-value pairs, comma separated value (CSV) files or some proprietary format, which requires an additional ETL step to import the data into ProM, which we want to avoid;

Ease of use The plugin should be straightforward to use with sensible defaults.

2.2 Functionality

Our plugin is started from within ProM and expects only a Petri net object as an input (which can easily be imported in ProM from many available file formats, such as PNML).

Fig. 1 guides the reader through the various steps of using the "Generate Event Log from Petri Net"-ProM plugin. After ProM is opened and a Petri

net model has been imported, this Petri net is used as an input object for the plugin, which can then be invoked. Fig. 1(a) shows the first configuration panel, and allows one to configure the following general simulation options:

Simulation method Available methods are: random generation, complete generation or grouped path (distinct) generation. The first method randomly executes enabled transitions (configurable with weights, see below) in the Petri net until an end condition is reached. The second method performs a complete exploration of the Petri net state graph to generate all possible traces (bounded by the "Maximum times marking seen" option, see below), while the final method ensures the generation of distinct traces, i.e. the same event log trace cannot be utilized twice.

Number of generated traces Amount of traces to generate (not available for complete generation).

Mininum/Maximum traces to add for each generated sequence How many "copies" of each generated trace should be added to the event log. The number is uniformly chosen between the minimum and maximum. Default is one (1) for minimum and maximum values, so the event log ends up containing exactly the amount of traces as configured by the "Number of generated traces"-option (for random and grouped path generation).

Maximum times marking seen How many times the same marking may be seen within a trace. This option bounds the amount of times a loop will be followed in the process model.

Only include traces that reach end state The simulation of a trace ends whenever there is no enabled transition which can be executed anymore (enabled traces might be available but prohibited by the "Maximum times marking seen"-option). In this case, the sequence of transitions is converted to a sequence of trace events and added to the event log. When the "Only include traces that reach end state"-option is set, traces are only added when there is a token in a sink place (a place not containing outgoing arcs). Naturally, when the users supplies a workflow net, only one such sink place is present.

Only include traces without remaining tokens Same as above, but now, traces are only added when *all* the tokens in the final marking are in a sink place. The default option enables the latter two options, which serves as a sensible default in most use cases to generate "valid", expected traces.

The following configuration screen, shown in Fig. 1(b), allows to configure global trace timing options. This includes:

An "anchor point" A date/time stamp used as a point of reference in following options.

Trace movement Both "moving" and "fixed" traces are available. For fixed traces, the initial start time for each trace is set equal to the given

anchor point. For moving traces, the starting point of the first trace is set equal to the anchor point, but the starting point of each following trace is set to the end (complete time of final activity) of the previous trace.

Variance Average and standard deviation in seconds to add variance to the starting point of each trace. This allows e.g. for partial overlap or volatile, randomized starting points.

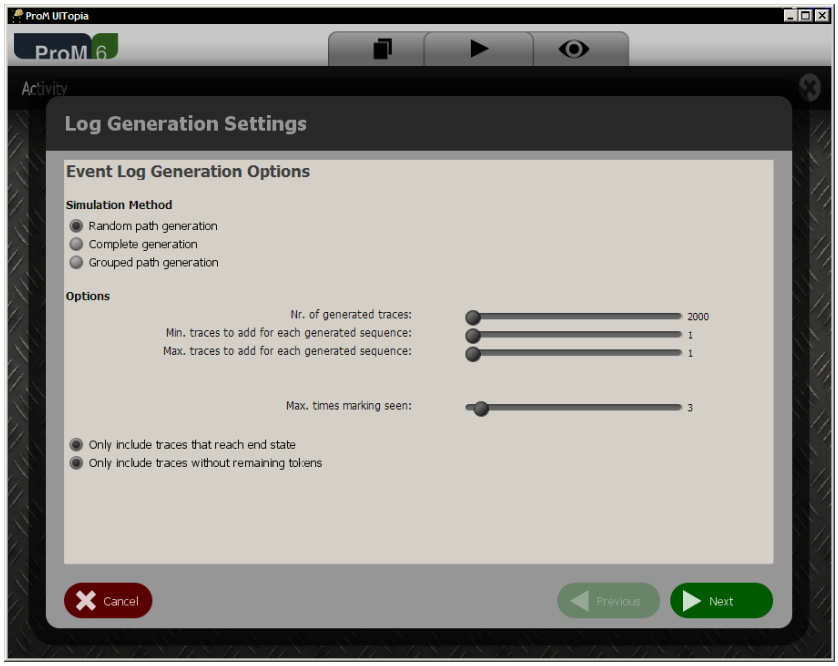
The next configuration screen in the wizard, depicted in Fig. 1(c), allows to map each Petri net transition to an activity label in the event log to be generated. Transitions can be set to invisible by leaving the label blank, and multiple transitions can be mapped to the same label. The reason why we allow users to do this is because not all Petri net modeling tools support saving Petri nets with an explicit distinction between invisible and visible transitions, and so that users can assign separate, understandable labels for duplicate transitions in a modeler (e.g. "activityDup_1" and "activityDup_2") while still mapping them to the same event log activity ("activityDup").

In addition, this screen also allows one to set transition "weights", which allow to drive the simulation towards choosing one enabled transition over others when multiple choices are available. The chance to select a transition $t \in T$ (all Petri net transitions) at each step is thus:

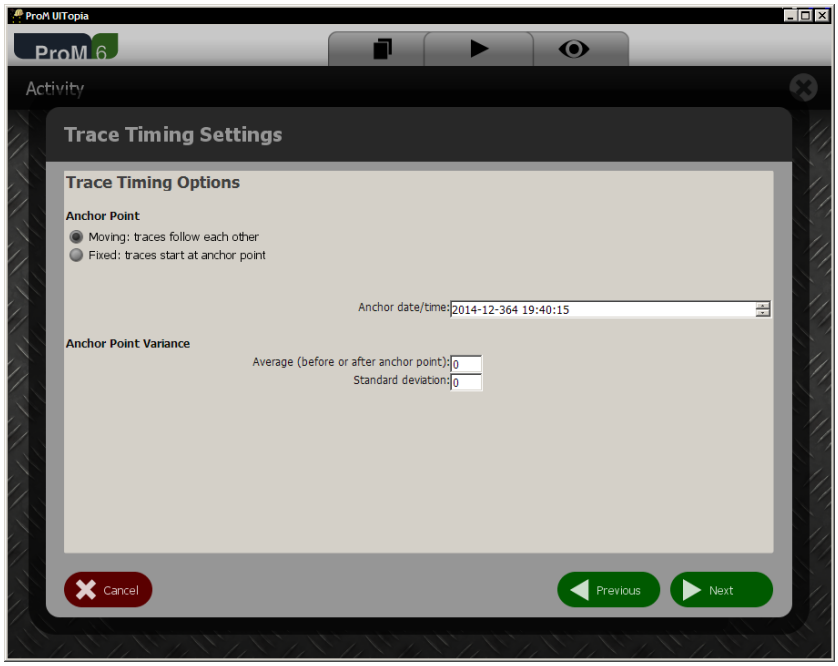
$$\begin{cases} 0 & \text{if } \neg enabled(t) \\ \frac{weight(t)}{\sum_{t \in T : enabled(t)} weight(t)} & \text{if } enabled(t) \end{cases}.$$

Finally, for each transition, the user can also display a "timings"-window (see Fig. 1(c)) which allows one to set averages and standard deviations for the lead times (duration) and idle times (waiting time before an activity start) of activities. The standard option inserts a fixed idle time of one minute between each activity and only generates a "complete" event for each activity (atomic activities), which is sufficient to generate event logs which can be used by the multitude of miners and other analysis techniques. Finally, for each transition, the user can also display a "resources"-window (see Fig. 1(d)) which allows one to configure the possible resources executing this activity, together with weightings. This is beneficial for users wanting to generate event logs to be used in social analysis scenarios.

After finishing the configuration, the generation process is started. Once done, the generated event log is opened in ProM (see Fig. 1(e)), where it can be further analyzed or exported to disk, using not only the XES file format but any format for which a ProM exporter is available. Fig. 1(f) shows a dotted chart analysis of the generated event log, illustrating at the same time the time variance induced in the generated log.

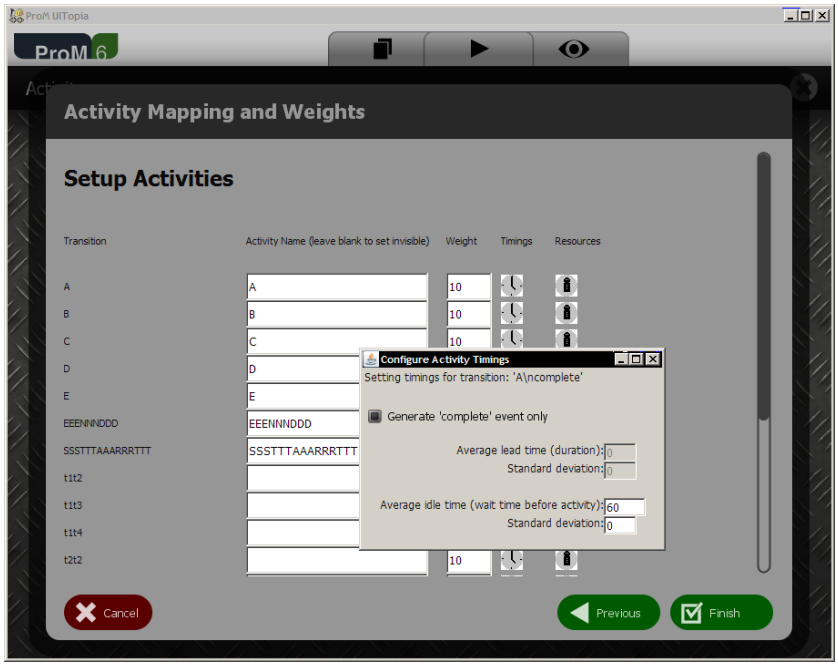


(a) Configuring simulation options.

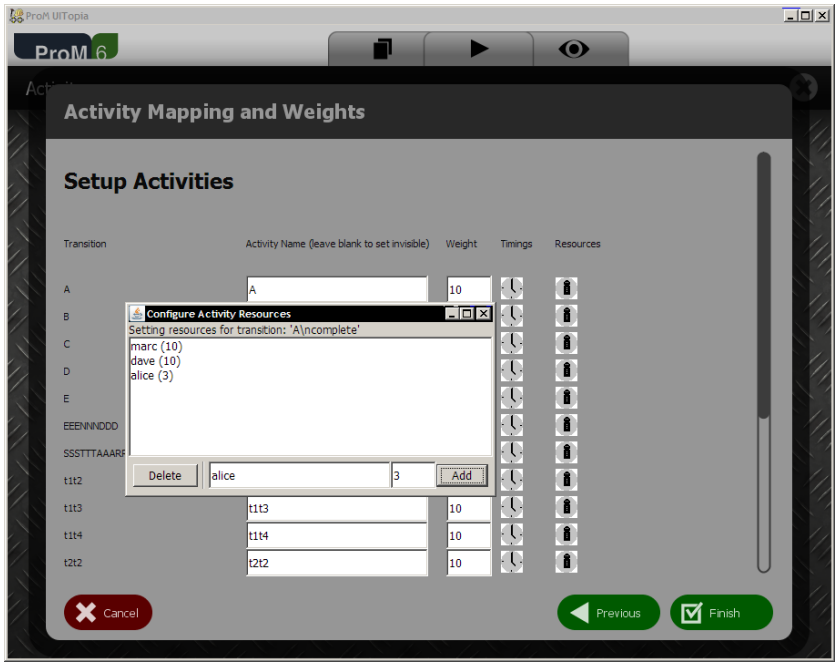


(b) Configuring global trace timings.

Fig. 1. Collection of screen shots depicting the various steps of the developed plugin.



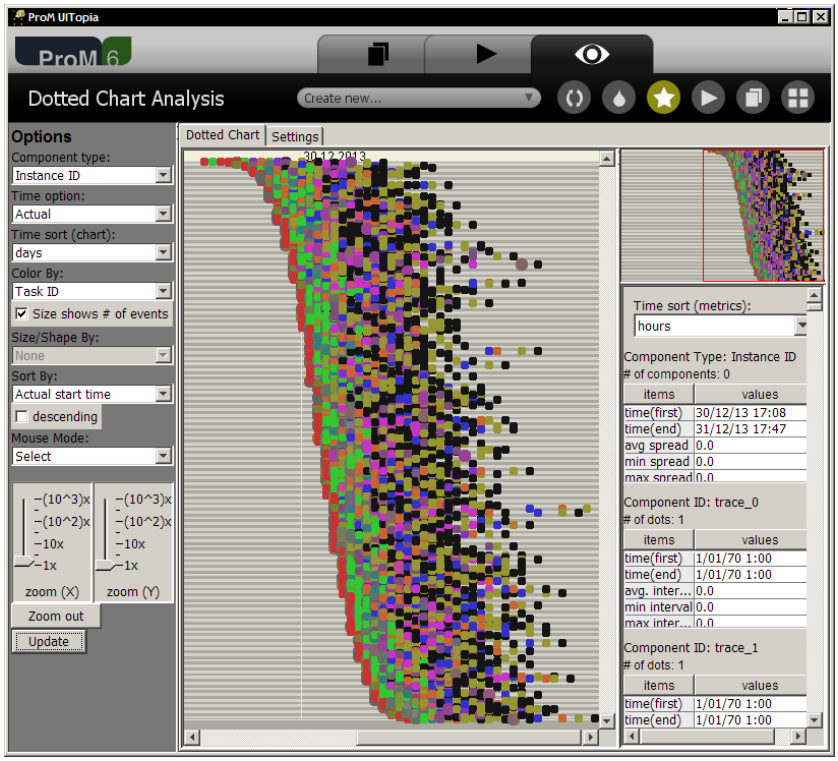
(c) Configuring activities and activity timings.



(d) Configuring activity resources.

Fig. 1. (continued) Collection of screen shots depicting the various steps of the developed plugin.

(e) Generated event log opens in ProM



(f) Analyzing the generated event log using dotted charts.

Fig. 1. (continued) Collection of screen shots depicting the various steps of the developed plugin.

2.3 Architecture

Our tool is build on top of the ProM framework and thus re-uses components of the latter. In particular, handling of Petri net models and event log concepts is based on ProM's internal models, as to maximize interoperability with existing plugins. Configuration screens are shown using ProM's default widgets.

To simulate traces from Petri nets, we make use of internally developed components, rather than relying on external libraries. Our plugin does not utilize any native components, and thus runs on every platform where Java and ProM are available. A full class overview is out of scope, but source code is available for inspection and modification by interested parties (see subsection "Installation" hereunder).

Data interchange with other tools is provided using ProM's standard import/export mechanisms. That is, Petri nets are loaded in using ProM's available file readers, and generated event logs are opened natively in ProM until the user is ready to export them. This also allows users to easily apply post-generation steps such as event log noise generation by means of other plugins.

2.4 Use Cases

As stated in the introduction, we believe our tool to be useful to scholars, practitioners and students working in the field of process mining. The authors have already utilized the presented plugin, both in research and educational settings. Some common use cases benefiting from generated synthetic logs include:

- Performing empirical experiments using a controlled data set;
- Constructing example data sets to use for educational purposes;
- Constructing illustrative data sets to present to usefulness of analysis techniques within a business context, where real-life data is not immediately available (due to technical or other reasons, e.g. privacy concerns).

2.5 Comparison

This section briefly compares our developed plugin against other tools and techniques. We hereby keep in mind the core objectives of our tool, offering a straightforward method to generate event logs from a Petri net in an environment familiar to process miners.

CPN Tools CPN Tools is the most widespread and advanced tool for editing, simulating and analyzing coloured Petri nets [2][2]. Although the latest versions have made it easier to model a simple (non-coloured) Petri net, the particular user interface of this tool comes with a rather steep learning curve. In addition, performing a repeated simulation run which can be outputted to an event log requires some setting

[2] See `http://cpntools.org`.

up and extract-load-transform steps using ProMimport [7, 8]. Plugins have also been made available in ProM which allow for more straightforward generation of event logs from CPN models in ProM [9], though knowledge of CPN modeling is still assumed (furthermore, in ProM 6.3 and CPN Tools 4.0—the latest stable releases of both—the simulation plugins crash due to not being able to handle the "real" data type of CPN models). By our plugin, we try to offer a simple alternative to CPN Tools when the ultimate goal is the generation of an event log given a Petri net.

Process Log Generator The Process Log Generator is developed by Burattin et al. [3], and allows users to generate random business processes by only specifying some simple parameters. The tool allows one to generate an event log from the generated process model, but does not allow to make modifications to the randomized process model or load in a Petri net designed in another modeler.

Petri net modeling Tools Inlcuding ARIS, FileNet Designer, FLOWer, PIPE, Protos, Renew, WoPeD, Yasper and other tools mentioned by the Petri Nets Tool Database [10]. A plethora of Petri net oriented modeling tools exist, with PIPE [11], Renew [12], WoPeD [13] and Yasper [14] being free, open and favored offerings by the process mining community. Many of these tools also allow to perform Petri net analysis, soundness verification, and to play the Petri net "token game". However, although many of these tools fully support the semantics of Petri net-based execution, they lack support to generate an event log as a whole using some configurable parameters. The purpose of our plugin is not to replace these tools, as they are still to be used to model (and verify) the Petri net from which an event log should be generated.

Other (Petri net oriented) simulation tools Including Arena, FileNet Simulator, GPenSIM, HiPS, Petri .NET Simulator, Stateflow, TINA. Other general purpose tools focus less on the modeling of Petri nets, but more on the simulation perspective. However, by "simulation", the statistical analysis of Petri net models is meant, rather than simulating (and saving) execution traces. Users are able to retrieve far reaching reports concerning utilization rates and mean durations, but no historic collection of traces can be saved. Furthermore, many of these tools are advanced and come with a steep learning curve.

Spreadsheets and other custom approaches Users also resort to using spreadsheets (with macros and formulas) or programming scripts against the OpenXES API to generate synthetic, randomized event logs. However, the usability of such techniques is less proven, as there now is no "true" process model driving the generation of traces. Furthermore, as spreadsheet-based approaches lack the capability to output a native event log, they also have to be converted to another format in a post-step.

2.6 Installation

Binary and source downloads, together with installation instructions can be retrieved at the following URL: `http://processmining.be/loggenerator/`. The plugin is open source and offered free of charge.

3 Conclusion

In this tool report, we have present a ProM plugin which allows for straightforward event log generation, using token-based simulation driven by Petri net models. The tool is deliberately implemented in ProM to allow for easy usage by process miners, is agnostic of the modeling tool used to build the Petri net, allows to configure various simulation options and focuses on easy of use.

A drawback of the tool is that it currently only support event log generation from Petri net models, but given this representation's popularity within the area of process mining and the availability of conversion plugins which allow to convert many other representational forms to a Petri net, this is not a limiting problem in practice. In future version of the tool, however, we plan to incorporate ways to handle more advanced constructs, such as inhibitor and reset arcs, in Petri nets. Other plans for future work incorporate: improving the look and feel of the user interface, allowing to save and load simulation settings, selecting custom distributions for random sampling of transitions, resources and timings.

Acknowledgements.

We would like to thank the KU Leuven research council for financial support under grant OT/10/010 and the Flemish Research Council for financial support under Odysseus grant B.0915.09.

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Jensen, K., Kristensen, L.M., Wells, L.: Coloured petri nets and cpn tools for modelling and validation of concurrent systems. International Journal on Software Tools for Technology Transfer **9**(3-4) (2007) 213–254
3. Burattin, A., Sperduti, A.: Plg: A framework for the generation of business process models and their execution logs. In Muehlen, M., Su, J., eds.: Business Process Management Workshops. Volume 66 of Lecture Notes in Business Information Processing. Springer Berlin Heidelberg (2011) 214–219
4. van der Aalst, W.M.P., van Dongen, B.F., Günther, C.W., Rozinat, A., Verbeek, E., Weijters, T.: Prom: The process mining toolkit. In de Medeiros, A.K.A., Weber, B., eds.: BPM (Demos). Volume 489 of CEUR Workshop Proceedings., CEUR-WS.org (2009)

5. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The prom framework: A new era in process mining tool support. In Ciardo, G., Darondeau, P., eds.: ICATPN. Volume 3536 of Lecture Notes in Computer Science., Springer (2005) 444–454
6. Murata, T.: Petri Nets: Properties, Analysis and Applications. In: Proceedings of the IEEE. Volume 77. (April 1989) 541–580
7. Medeiros, A.K.A.D., Gunther, C.W.: Process mining: Using cpn tools to create test logs for mining algorithms. In: Proceedings of the Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools. (2005) 177–190
8. Nakatumba, J., Westergaard, M., Aalst, W.: Generating event logs with workload-dependent speeds from simulation models. In Bajec, M., Eder, J., eds.: Advanced Information Systems Engineering Workshops. Volume 112 of Lecture Notes in Business Information Processing. Springer Berlin Heidelberg (2012) 383–397
9. Westergaard, M., van Dongen, B.: Keyvaluesets: Event logs revisited (2013)
10. World, P.N.: Complete overview of petri nets tools database
11. Dingle, N.J., Knottenbelt, W.J., Suto, T.: Pipe2: A tool for the performance evaluation of generalised stochastic petri nets. SIGMETRICS Perform. Eval. Rev. **36**(4) (March 2009) 34–39
12. Kummer, O., Wienberg, F., Duvigneau, M., Schumacher, J., Kohler, M., Moldt, D., Rolke, H., Valk, R.: An extensible editor and simulation engine for petri nets: Renew. In Cortadella, J., Reisig, W., eds.: Applications and Theory of Petri Nets 2004. Volume 3099 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2004) 484–493
13. Eckleder, A., Freytag, T., Karlsbad, N.A.: Woped 2.0 goes bpel 2.0
14. van Hee, K., Oanea, O., Post, R., Somers, L., v an der Werf, J.M.: Yasper: a tool for workflow modeling and analysis. 2010 10th International Conference on Application of Concurrency to System Design **0** (2006) 279–282

FACULTY OF ECONOMICS AND BUSINESS
Naamsestraat 69 bus 3500
3000 LEUVEN, BELGIË
tel. + 32 16 32 66 12
fax + 32 16 32 67 91
info@econ.kuleuven.be
www.econ.kuleuven.be