Towards an Open and Extensible Business Process Simulation Engine

Luciano García-Bañuelos* and Marlon Dumas

Department of Computer Science
University of Tartu, Estonia
{luciano.garcia,marlon.dumas}@ut.ee

Abstract. This paper outlines the architecture and initial proof-of-concept implementation of an open and extensible business process model simulator based on CPN tools. The key component of the simulator is a transformation from BPMN process models to CPNs. This transformation is structured as a set of templates that can be extended and modified by developers in order to incorporate new functionality into the simulator. The paper illustrates how this templating mechanism can be used to capture different types of tasks and resource allocation policies.

1 Introduction

Business process simulation is a widely used technique for analyzing business process models with respect to performance metrics such as cycle time, cost and resource utilization. Many commercial business process modeling tools incorporate a simulation component, e.g. TIBCO Business Studio, IBM Websphere Business Modeler (WBM), ARIS, FileNet and Protos [5]. However, these process simulators have two architectural limitations:

1. Only models designed with the tools themselves can be simulated.
2. No extensibility mechanism is provided to add new features or change the pre-built simulation and reporting options.

The first limitation stems from two factors. Firstly, the simulation component is generally hidden inside the tool, meaning that it does not expose an Application Programming Interface (API). Secondly, there is a long-standing problem of lack of business process model interoperability. The Business Process Modeling Notation (BPMN) – particularly its upcoming version 2 – attempts to address this issue by providing a standard business process meta-model with an interoperable XML serialization. While BPMN does not define any simulation parameters (e.g. arrival times and branching probabilities), it includes extensibility mechanisms that allow one to add such details.

The second limitation is to a large extent connected to the first one: since the tools do not offer an explicit interface for their simulation component, it

* On leave from Autonomous University of Tlaxcala; work funded by the ERDF through the Estonian Centre of Excellence in Computer Science.

is not possible to plug-in additional functionality into it. This limitation has been raised in recent work [7]. Motivated by simulation requirements found in supply chain management, the authors introduce an extensibility mechanism into IBM WBM. The idea is to allow developers to attach scripts to processes and tasks. Six types of scripts are supported: pre-processing scripts (executed before a task/process is activated), post-processing scripts, delay scripts (to introduce waiting times), cost, revenue and duration scripts. However, this mechanism is limited – it does not allow one to extend the execution semantics that drives the simulation engine. For example, one cannot apply such extensibility mechanism in order to simulate advanced resource allocation patterns [10] nor to capture control-flow connectors beyond those offered by the modeling tool.

In this paper, we outline the architecture and current implementation status of OXProS - an Open and Extensible Process Simulator for BPMN. OXProS provides a RESTful service interface allowing third-party applications to submit BPMN process models for simulation. Simulation parameters are incorporated into the process model using BPMN's extensibility mechanism. OXProS uses CPN Tools as its underlying execution environment. In other words, BPMN models are translated into CPNs for simulation purposes. This translation is based on a templating mechanism that enables developers to extend OXProS by adding new templates or modifying existing ones.

In the rest of the paper we present the architecture of OXProS and its template-based extensibility mechanism. We present the current implementation status of OXProS and a roadmap for future work.

2 Architecture

Figure 1 depicts the architecture of OXProS. In this figure, the yellow boxes represent OXProS components, the light-blue boxes represent CPN Tools and associated components, and the white boxes correspond to third-party software.

At the architectural level, OXProS is structured as a set of XML over HTTP Web services, designed according to the principles of RESTful service architectures [3]. A simulation is treated as a resource that can be created/started through a POST operation. Subsequently, the status of the simulation and its output can be retrieved using GET operations. As a shortcut, it is possible to create a simulation model and retrieve its results through a single POST operation.

OXProS accepts either BPMN process models (enhanced with simulation attributes) or CPN models. If a BPMN model is submitted, it is transformed into a CPN model and passed on to the CPN simulation service. At present, there is no standard serialization of BPMN. It is expected that the upcoming BPMN 2.0 standard will have its own XML serialization format. In the meantime, the BPMN simulation service assumes that the input models are serialized using the XML Process Definition Language (XPDL) version 2.1 [1].

The services provided by OXProS are implemented using Java Servlets. The CPN simulation servlet is built on top of Access/CPN [13] – a tool that enables

programmatic access to the CPN Tools Simulator. Since Access/CPN is implemented as a set of OSGi bundles/components, its implementation relies on the HTTP Service provided by Equinox – the Eclipse OSGi implementation[1].

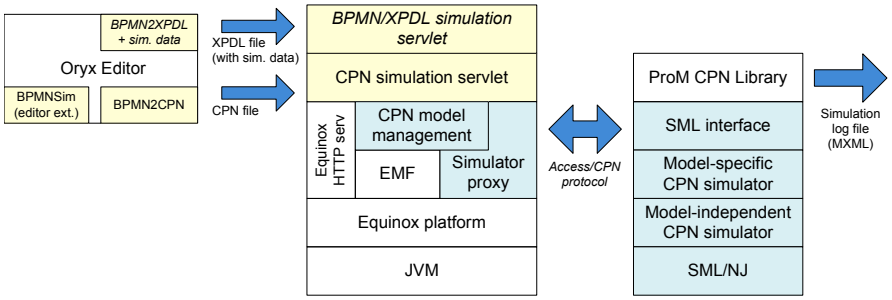


Fig. 1. OXProS architecture

When the CPN simulation servlet receives a simulation model, it invokes Access/CPN to check the submitted CPN and starts the simulation on the background simulator daemon. Upon completion of the simulation, an MXML log file is generated from the simulation output. The produced MXML log file can be analyzed off-line using the ProM framework[2] (e.g. to extract key performance indicators). In order to generate MXML log files, we make use of the ProM CPN Library [6].

In the current prototype, BPMN processes are modeled with the Oryx Editor[3]. We extended the Oryx front-end to allow users to add simulation parameters into a BPMN process model. This Oryx extension is called BPMNSim. Moreover, we extended the Oryx back-end with a module that generates CPNs from Oryx's internal representation of process models. In the future, it is expected that Oryx will support generation of standard BPMN models so that these models can be submitted to the BPMN simulation servlet.

3 Template-based Generation of CPNs

The key component of OxProS is the transformation from extended BPMN models to CPNs. This transformation is designed using a templating approach. To illustrate the transformation, we consider the simplified "teleclaims" process of an insurance company, presented in Figure 2. This process handles insurance claims made by phone. The process is supported by staff in a call center and in a claims handling department.

In the basic form, a BPMN process diagram consists of events (circles), activities (rounded rectangles) and gateways (diamonds). Events denote the start,

[1] http://www.eclipse.org/equinox/server
[2] http://www.processmining.org
[3] http://oryx-editor.org

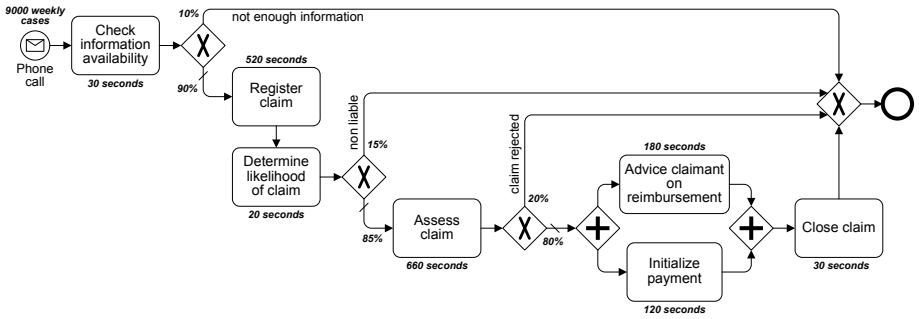


Fig. 2. Insurance claim handling process (adapted from [12])

the end of a process case or something that happens during the process execution. In the running example, we used a "message start event" to specify that a case starts with the reception of a "Phone call". Activities denote work that must be done, for example, an agent in the call center executes the task "Check information availability". Gateways are routing constructs. The two basic types of gateways are AND gateways (marked with a "+" symbol) and XOR gateways ("X" symbol). A gateway is either to be a split gateway if it has multiple outgoing flows or a join gateway if it has multiple incoming flows. An XOR-split is a decision point, meaning that one outgoing path is taken according to the result of the evaluation of a logical condition. For instance, in the running example, 10% of cases are rejected due to insufficient information. A XOR-join is a merging point. An AND-split starts two or more parallel threads. For instance, tasks "Advice claimant on reimbursement" and "Initialize payment" are executed in parallel. Finally an AND-join synchronizes parallel threads.

In order to simulate a BPMN process model, a number of simulation parameters need to be specified. These include: (i) the arrival rate of new cases and its associated distribution (e.g. exponential); (ii) the branching probabilities for each arc (flow) leaving an XOR-split, and a number of performance attributes attached to activities/events, such as time, cost and revenue. These attributes are generally represented using a mean value and a probability distribution (e.g. normal). A simulation model includes a number of *resource pools* denoting sets of resources. Each activity may be associated to a resource pool and at runtime, one resource from the pool is selected to perform the activity based on a resource allocation policy.

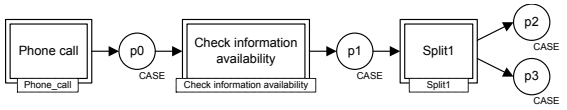
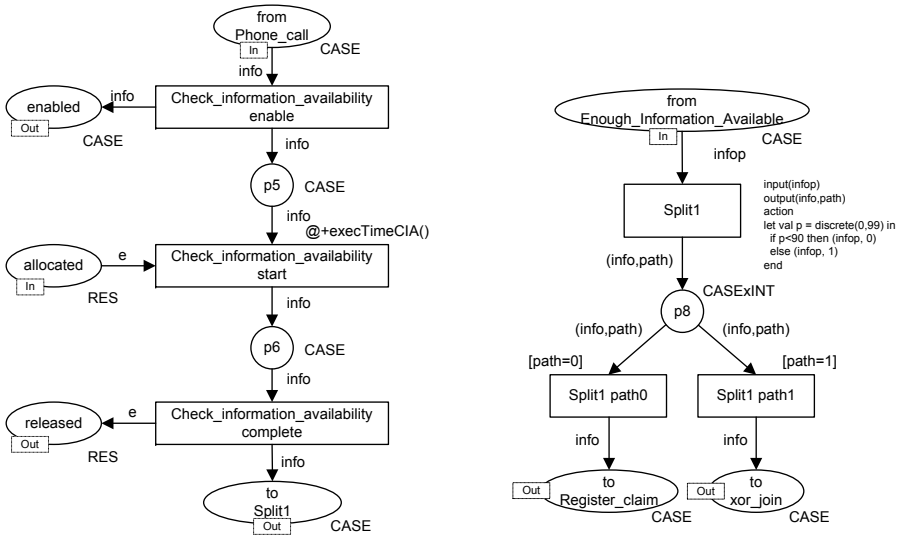


Fig. 3. Mapping of the root process

Given a BPMN model extended with simulation attributes, we generate a hierarchical CPN, with two top-level pages: one for the control flow perspective and the other for resource perspective. Figure 3 presents a partial view of the CPN model for the control-flow of the insurance claim process.



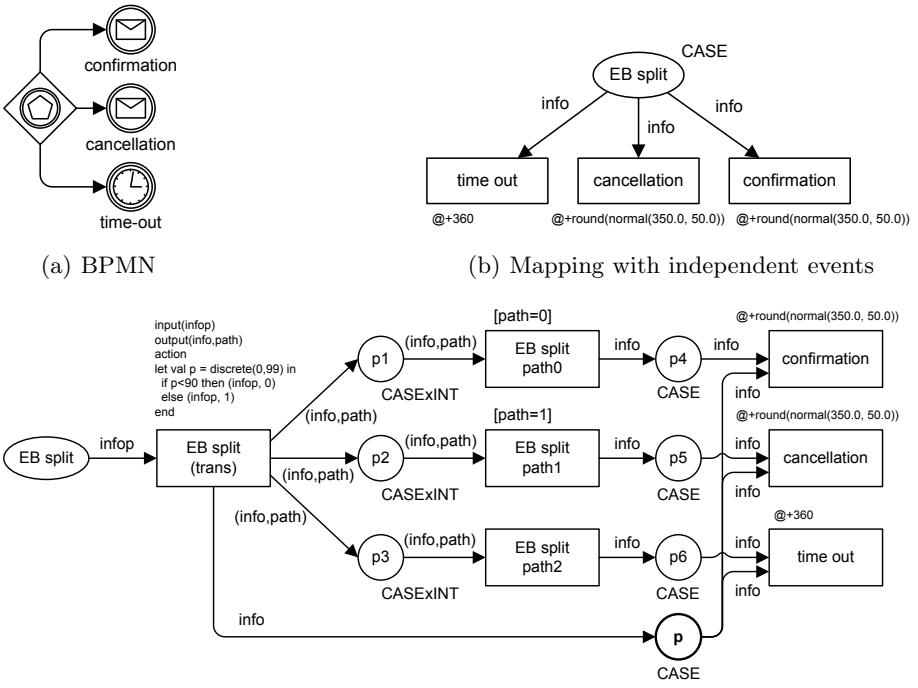
(a) Task (b) XOR (split) gateway

Fig. 4. Default mapping of tasks and XOR-split gateways

In order to make the mapping as modular as possible, every element in a process model is mapped to an individual page. For example, Figure 4(a) presents the page with the mapping of task "Check information availability" from the running example. It contains two places to synchronize the start and completion of the task, to model the control flow perspective. Additionally, the page includes three other places to handle the assignment of the task to a staff member for execution. A simulation would occur as follows. First, the task gets enabled and waits until a staff member takes it. The timing annotation on transition "Check_information_availability start" generates a random simulation time that is associated to the execution of the task. When the task is completed, the resource is released.

Let us now consider the case of data-based XOR split gateways. The corresponding CPN must handle the selection of a path according to a probability distribution. Figure 4(b) presents the page for the first XOR gateway in the running example. The transition "Split1" generates a random value between 0 and 99. For every path, there is a guarded transition which is activated according to the value of the variable "path". Thus, the transition "Split1" sets "path" to 0 in 90% of the cases and it sets "path" to 1 in the remaining 10% of the cases.

In addition to the XOR decision gateway discussed above (called *data-driven decision gateway*), BPMN features a so-called event-driven XOR decision gateway. The data-based and the event-based XOR gateways are similar in that they pass the flow of control along one of their outgoing paths only – thus, they represent a "choice". However, whilst in a data-based XOR gateway the choice is performed by the system based on boolean conditions, in the case of an event-based XOR gateway, the choice is made by the environment. Specifically, an



(a) BPMN

(b) Mapping with independent events

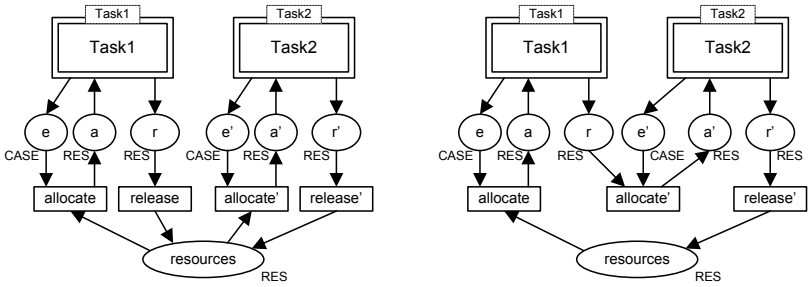
(c) Mapping with dependent events

Fig. 5. The BPMN Event-based XOR gateway

event-based XOR gateway is directly followed by two or more events. When the gateway is reached, the flow of control stops at that point until one of the events in question occurs. The first event to occur determines which path is taken. Figure 5(a) shows a typical scenario involving an event-based XOR gateway. In this example the process waits for either a confirmation or a cancellation message from the customer, or a time-out.

Multiple approaches can be adopted to simulate BPMN models with event-based XOR gateways. One approach is to capture it purely using delays: each event is annotated with a delay, that may be a fixed duration (in the case of time events with constant values), or a probability distribution (in the case of message events or time events with a dynamically-evaluated duration). Figure 5(b) illustrates this approach. Here, the delays of the two message events (cancellation and confirmation) follow a normal distribution with identical means and standard deviations. This mapping works under the assumption that the two events are independent.

However, in this example one would expect that these two events are not independent, but rather exclusive: if a confirmation message is received, no cancellation message will arrive and vice-versa. Figure 5(c) presents an alternative mapping that captures this exclusion dependency. In this case, the customer re-



(a) Role-based distribution (b) Chained execution

Fig. 6. Two templates for resource allocation

sponds with a confirmation message in 90% of the cases or with a cancellation message in 10% of the cases. The ML annotation on the first transition activates the path to the "time-out" event and to one of the other two paths (the confirmation path and the cancellation path). From that point on, there is a race between the time event and one of the two message events, reflecting the fact that only one of the two message events may eventually occur.

In order to implement this alternative mapping, one has to extend BPMN with the ability to express exclusion dependencies between events that are connected to a common event-based gateway. In the current BPMN standard, when two events are attached to an event-based gateway, it might be that either event may occur or that both events may occur (even though the second one to occur may be discarded). In the above example, the case where both messages are eventually received by the process is excluded. In other words, the events are not in a "race" but rather in an exclusion relation. While this exclusion relation between message events is not relevant for execution purposes, it is relevant for (stochastic) simulation since it means that their occurrence should be drawn from a common probability distribution. In OXProS, we rely on a non-standard extension of event-based decision gateways to capture this exclusion dependency.

In OXProS, CPN simulation models are generated using *templates*. A template takes as input a BPMN element of a given type, and produces a CPN page. Multiple templates can exist for a given type of BPMN element. Since each BPMN construct is mapped to a separate page, the mapping of a type of element (e.g. task, event-based XOR gateway) can be replaced by a different mapping without affecting other elements, so long as the inputs/output places of the page generated by the new mapping are compatible with those of the previous mapping.

Templates are also used for capturing resource management. For example, Figure 6 shows two templates for resource allocation. The first one – Figure 6(a) – corresponds to the "Role-based distribution" workflow resource pattern [10].[4] The idea behind this pattern is that at runtime the execution engine routes the

[4] For readability reasons, we omitted some details on the colored Petri nets in Figure 6.

task to the worklist of the resources that can perform a given task, e.g. staff members with a particular profile/role. Eventually, one of those resources will remove the workitem from worklist and perform it. This can be simulated with an ML code attached to the transition "allocate", and a timing annotation can be used to record the waiting time. When the task is completed, the resource is sent back to the resource pool by transition "release". Existing business process simulation tools generally implement this approach. However, there exist other resource allocation patterns. As an example, consider the CPN shown in Figure 6(b) that captures the so-called "Chained execution" pattern. In this case, the two subsequent tasks are assigned to the same resource. Both of these templates are provided in OXProS, and developers may introduce additional alternative templates. When a BPMN model is submitted to OXProS for simulation, the request may refer to a *simulation profile* that determines which template should be used for which BPMN construct. If no profile is specified, the default profile is used.

4 Related work

Many commercial business process modeling tools incorporate a simulation component, e.g. TIBCO Business Studio, IBM Websphere Business Modeler, ARIS, FileNet and Protos, among others. Jansen-Vullers and Netjes [5] survey a number of process simulation tools and evaluate their suitability with respect to a number of requirements. They conclude that no tool covers all the requirements, and that CPN tools is a suitable option in terms of expressiveness but that it may be too complex for use by business analysts.

A translation of Protos simulation models to CPN models is presented in [4]. This translation is rather straightforward because Protos models have many commonalities with Petri nets. The authors also describe an extension of their translation to handle configurable process models. A configurable process model provides an integrated view of multiple process model variants. The tool presented in [4] is geared towards comparing the performance of multiple process model variants captured in a given configurable process model.

Rozinat et al. [8] present an approach to mining simulation models from event logs. The idea is to automatically generate a process model, represented as a CPN, which will behave in a similar way as the process model that generated the original event log. Depending on the richness of the event log, the resulting CPN may cover not only the control-flow perspective, but also the data perspective (e.g. data attributes and branching conditions), the organizational perspective (e.g. roles, resources) and the performance perspective (e.g. distribution of execution times). The authors follow the approach of mapping each activity in the process to a separate sub-page, an idea which is also followed in OXProS.

CPN tools has been used for generating synthetic logs to be used in the design and testing of Process Mining algorithms [6]. In contrast to real-life data, the log generated by simulation is free of noise and will include the perspectives that

are important for tuning a given mining algorithm. In [6], the authors outline a set of ML functions to extend CPN tools in order to generate logs in the MXML format. This is the format used by several process mining tools. OXProS follows the idea of representing the simulation output in MXML and reuses the library of MXML generation functions defined in [6].

In [11], the authors argue that current approaches to model resource allocation for process simulation are not realistic. For instance, scenarios where the same resource is assigned to multiple processes (and thus needs to split its time between them) are not supported by existing tools. Also, existing tools fail to take into account that people work in batches and that they divide their time into discrete intervals ("chunks") that they allocate to different types of tasks. They present a novel approach to capturing simulation models in which each resource is captured as a separate CPN page. The CPN page of a resource captures the resource's lifecycle. Therefore, each resource (or each resource class) may have a different lifecycle. While this approach leads to more realistic resource models, one has to note that implementing such an approach requires significant input from the modeler, since the modeler has to provide additional information for each resource (class) and this information may not be available in some cases.

In line with the above body of research, we adopt CPNs as a basis for business process model simulation. Unlike this body of work however, we adopt BPMN as the notation for modeling business processes, since it is a widely-adopted standard that strikes a tradeoff between ease-of-use for business analysts and expressiveness. While some of the previous work can be easily adapted to support the simulation of BPMN models, BPMN brings in certain specificities that warrant further study. Specifically, BPMN has a rich set of event types and events can be used in different settings (e.g. event-based gateways, error events). We have shown in this paper that there are multiple possible approaches for simulating process models with event-based gateways and message events. Each of these approaches strikes a different tradeoff between the amount of input data required for simulation and the precision with which the simulation reproduces the real-world phenomenon. The multiplicity of possible approaches justifies the need for an extensible architecture which is one of the driving requirements of OXProS.

5 Outlook

The current implementation of OXProS supports a restricted subset of BPMN – XOR/AND gateways, tasks, plain events and start message events. Ongoing work aims at extending the coverage of BPMN by adding templates incrementally. The majority of these templates will be designed on the basis of the BPMN to plain Petri nets transformation outlined in [2].

In order to validate the suitability of the template-based extensibility mechanisms, we then plan to simulate process models from the logistics domain, where resources (e.g. trucks) have capacity and speed constraints that cannot be captured using commercial business process simulation tools.

The current OXProS architecture is only able to generate MXML simulation logs and leaves it up to the user to analyze these raw logs. In future, we plan to incorporate analytics services into the OXProS architecture in order to compute key performance indicators from the simulation logs. Another avenue for future work is to allow OXProS to take as input not only process models with simulation attributes, but also process execution logs, in order to simulate process models based on real past executions and starting from a given state [9].

References

1. Workflow Management Coallition. Process Definition Interface – XML Process Definition Language, October 2008.
2. R. Dijkman, M. Dumas, and C. Ouyang. Formal Semantics and Analysis of BPMN Process Models. *Information and Software Technology*, 50(12):1281–1294, 2008.
3. Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.
4. F. Gottschalk, W. M. P. van der Aalst, M. H. Jansen-Vullers, and H. M. W. Verbeek. Protos2CPN: using colored Petri nets for configuring and testing business processes. *International Journal on Software Tools for Technology Transfer*, 10(1):95–110, December 2007.
5. M.H. Jansen-Vullers and M. Netjes. Business Process Simulation – A tool survey. In *Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, 2006.
6. A. K. Alves De Medeiros and C. W. Günther. 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*, pages 177–190, 2005.
7. C. Ren, W. Wang, J. Dong, H. Ding, B. Shao, and Q. Wang. Towards a Flexible Business Process Modeling and Simulation Environment. In *WSC '08: Proceedings of the 40th Winter Simulation Conference*, pages 1694–1701, 2008.
8. 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, December 2007.
9. A. Rozinat, M.T. Wynn, W.M.P. van der Aalst, A.H.M. ter Hofstede, and C.J. Fidge. Workflow Simulation for Operational Decision Support. *Data & Knowledge Engineering*, 68(9):834–850, 2009.
10. N. Russell, W.M. P. van der Aalst, A.H. M. ter Hofstede, and D. Edmond. Workflow Resource Patterns: Identification, Representation and Tool Support. In *17th International Conference on Advanced Information Systems Engineering (CAiSE), Porto, Portugal, June 13-17, 2005*, pages 216–232. Springer, 2005.
11. W.M.P. van der Aalst, J. Nakatumba, A. Rozinat, and N. Russell. Business Process Simulation: How to get it right? Technical Report BPM-08-07, BPMcenter.org, 2008.
12. W.M.P. van der Aalst, M. Rosemann, and M. Dumas. Deadline-based Escalation in Process-Aware Information Systems. *Decision Support Systems*, 43(2):492–511, 2007.
13. Michael Westergaard and Lars Kristensen. The Access/CPN Framework: A Tool for Interacting with the CPN Tools Simulator. *Applications and Theory of Petri Nets*, pages 313–322, 2009.