THE BUSINESS PROCESS SIMULATION STANDARD (BPSIM): CHANCES AND LIMITS

Ralf Laue
Department of Computer Science
University of Applied Sciences of Zwickau
Dr.-Friedrichs-Ring 2a, 08056 Zwickau, Germany
Ralf.Laue@fh-zwickau.de

Christian Müller
Faculty of Business, Computing, Law
Technical University of Applied Sciences Wildau
Hochschulring 1, D-15745 Wildau, Germany
christian.mueller@th-wildau.de

KEYWORDS

Event driven simulation, business processes, process analysis, Business Process Modeling and Notation (BPMN), Business Process Simulation Interchange Standard (BPSim)

ABSTRACT

This paper provides a critical analysis of the BPSim standard, a specification by the Workflow Management Coalition. The aim of this standard is to make it possible to exchange simulation models between different modeling and simulation tools. We discuss the expressiveness of BPSim model and come to the conclusion that it will be sufficient for certain cases, but also lacks some important features.

INTRODUCTION

Business processes models are usually specified in graphical languages. The most popular standard for such a language is Business Process Model and Notation (BPMN) [BPMN 2013].

Figure 1 BPSim Contributors

For simulation purposes, the models have to be enriched by additional information and transformed into formal specifications that can be processed by a simulation tool [Anthony Wallner et. al. 2006, Raimar Scherer 2011]. The Business Process Simulation Interchange Standard (BPSim) is a BPMN extension for process simulation. It was developed by some industrial actors (Fig. 1) and published as a standard specification [BPSim 2013, BPSim 2014].

Not all products of the contributors are fully supporting the BPSim specification. Known implementations were provided by Trisotech, Lanner, Sparx and jBPM.

A BPSim simulation engine is not only an extension of an BPMN engine, because the aim of a BPMN engine is process automation and not simulation. Such an automation engine must store its data persistently in a database. Simulation runs must be fast, hence a simulation engine should store the data in internal memory. For this reason, implementing a simulation engine in a BPMN suite requires considerable effort.

In this paper, we discuss the main ideas of BPSim as well as its chances and limits.

BPMN DIAGRAMS AND SERIALIZATION

Before we start discussing BPSim as an extension of BPMN2 (version 2.0 is the current version of the BPMN standard), we will describe the basic ideas of BPMN2. Its aim is the modeling of business processes for documentation and automation purposes. The standard [BPMN] defines the graphical representation of models, its semantics and an XML-based serialization format. In Fig. 2, we show a BPMN diagram that we will use as an example throughout this paper. First, a decision task is executed. At a subsequent gateway, the process path splits depending on the outcome of the decision.

The basic (simplified) XML file structure for this process fragment is shown in Fig. 3. A definitions element contains the required resources and processes.

Proceedings 30th European Conference on Modelling and Simulation ©ECMS Thorsten Claus, Frank Herrmann, Michael Manitz, Oliver Rose (Editors)
ISBN: 978-0-9932440-2-5 / ISBN: 978-0-9932440-3-2 (CD)

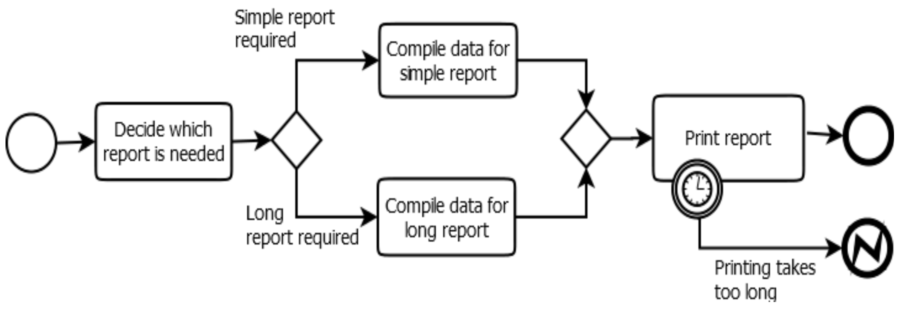

Simple report required
Compile data for simple report
Decide which report is needed
Print report
Long report required
Compile data for long report
Printing takes too long

Figure 2 Example Process

The *process* element includes its events, tasks, gateways and sequence flow arcs. The XML file contains also information about resources and data that is not shown in the diagram: The *resource* tag describes the performer of a task and has a model-wide scope. In contrast to this, the *property* element belongs to a certain process. It describes a variable with a scope of a process instance. It is also possible to use a so-called datastore for variables with a model-wide scope (not just referring to a single process execution). In our example, we did not use this feature.

The decision logic is modeled at the outgoing arcs of the gateway by 2 expressions, deciding which arc is used.

```
<definitions xmlns="...">
  <resource id="resource:id" />
  <process id="process_id" >
    <property id="property_id" name="report"/>
    <startEvent id="start_id" />
    <sequenceFlow sourceRef="start_id" targetRef="task_id"/>
    <task id="task_id">
      <performer id="performer_id" >
        <resourceRef>resource_id</resourceRef>
      </performer>
    </task>
    <sequenceFlow sourceRef="task_id" targetRef="gatw_id"/>
    <exclusiveGateway id="gatw_id"   />
    <sequenceFlow sourceRef="gatw_id" targetRef="task2_id">
      <conditionExpression>
        <![CDATA[decision == 'simple_report';]]>
       </conditionExpression>
    </sequenceFlow>
   <sequenceFlow sourceRef="gatw_id" targetRef="task3id">
      <conditionExpression>
        <![CDATA[decision == 'long_report';]]>
       </conditionExpression>
    </sequenceFlow>
    …..
  </process>
  <bpmndi:BPMNDiagram><!--Refers to graphical
diagram layout--></bpmndi:BPMNDiagram>
</definitions>
```

Figure 3

BPSIM EXTENSION

BPMN supports extensions for different purposes. One such extension is the Business Process simulation Interchange Standard (BPSim) [BPSim 2013, BPSim 2014] for simulating business processes. Using the bpsim namespace (http://www.bpsim.org/schemas/1.0), it adds additional information to an XML serialization in the BPSimData node (Fig. 4).

A BPSim model is organized in different simulation scenarios with start, duration, seed and replication values (Fig 5). A parameter for a warm-up period is missing, but some vendors have extended their implementation by this parameter. For working with variations of scenarios, the inheritance of scenarios is supported.

```
<definitions>
  <resource />  <process /> ..
  <bpmndi:BPMNDiagram> </bpmndi:BPMNDiagram>
  <relationship type="BPSimData">
     <extensionElements>
        <bpsim:BPSimData xmlns:bpsim=
                 "http://www.bpsim.org/schemas/1.0">
        </bpsim:BPSimData>
     </extensionElements>
  </relationship>
</definitions>
```

Figure 4

It is possible to assign simulation parameters to BPMN nodes and arcs by means of an *elementParameter* in a scenario definition. E.g., it is possible to define a *durationParameter* for task nodes (Fig. 5).

Various time parameters (such as setup time or processing time) can be added to tasks. Control parameters allow to define how often / with which probability certain events or certain decisions occur. Resources can be defined and it is possible to assign resources to tasks. In addition, tasks can have priorities. Fixed costs and costs per unit can be assigned to both tasks and resources. All these attributes can be defined depending on calendar definitions, for example by specifying that the availability of resources depend on workdays or shifts.

All the parameters described above can be either fixed values, historical data series or defined as realization of random variables that follow a certain distribution. The standard allows to use 13 types of distributions (that can be parametrized) for this purpose. This is clearly positive – the authors are aware of several business process simulation tools that work with a too limited set of distributions.

```
<bpsim:Scenario id="default" name="Scenario" ….>
  <bpsim:ScenarioParameters
       replication="2" seed="999" … >
   <bpsim:Start>
     <bpsim:DateTimeParameter value="2016-01-01T00:00:00"/>
    </bpsim:Start>
  </bpsim:ScenarioParameters>
  <bpsim:ElementParameters elementRef="task_id">
    <bpsim:TimeParameters>
      <bpsim:ProcessingTime>
        <bpsim:DurationParameter value="PT1H"/>
      </bpsim:ProcessingTime>
    </bpsim:TimeParameters>
  </bpsim:ElementParameters>
</bpsim:Scenario>
```

Figure 5

In a BPMN model, properties and data objects are used to control the execution of the model instances. In addition, BPSim allows to add properties to each node and arc by means of *ElementParameters*. The relation between BPMN and BPSim parameters is not specified and depends on the simulation engine, e.g., it is not clear how a model should be interpreted if its BPMN2 parameters for resource usage contradict to the BPSim *ResourceParameters*.

For extending our model to a classical simulation model with a capacity of e.g. 10 resources and a requirement of e.g. 2 resources per task we can use the following parameters of Fig. 6 (all belonging to the BPSim namespace):

```
<ElementParameters elementRef="resource_id">
...<ResourceParameters>
…...<Quantity>
….......<NumericParameter value="10"/>
…...</Quantity>
...</ResourceParameters>
</ElementParameters>

<ElementParameters elementRef="task_id">
...<ResourceParameters>
…...<Selection>
….......<ExpressionParameter value=
            "bpsim:getResource('resource_id', 2)" />
…...</Selection>
...</ResourceParameters>
</ElementParameters>
```

Figure 6

This definition (as given in [BPSim 2014]), runs in the simulation engine from Lanner, but it contradicts to the standard specification ([BPSim 2013, Sect. 7.3]) that defines that *ResourceParameters* are not associated to a *task_id* but to the *performer_id*, which belongs to the task element.

For modeling a time schedule, the standard allows calendar-depended parameters. Unfortunately, this feature is not supported by all tools.

Parameters can be marked as *ResultRequest* (Fig. 7) in order to collect the results of a simulation run. This allows to ask for minimum, maximum and mean values (for example of costs or durations), sums (aggregated values, e.g., total time spent in a certain task) and for the number of occurrences (for example of a certain event).

```
<ElementParameters elementRef="task_id">
...<TimeParameters>
…...<WaitTime>
….......<ResultRequest>sum</ResultRequest>
…...</WaitTime>
…...<ProcessingTime>
….......<ResultRequest>sum</ResultRequest>
…...</ProcessingTime>
...</TimeParameters>
</ElementParameters>
```

Figure 7

CHANCES AND LIMITS OF BPSIM

A lot of simulation models for business processes have simple scenarios. For such cases, the BPSim approach (adding parameters to BPMN elements by means of the BPSim extension) works well: For these models its a great improvement that BPSim allows formulating simulation models independently from modeling tools and simulation engines.

However, in a practical test, we found that on the one hand, some tools implement only a subset of the standard. On the other hand, they provide useful (but proprietary) vendor extensions. E.g. the Trisotech modeler does not support the assignment of a resource parameter to a task. For running a simulation with resources, the model must be changed by hand in text editor. Hopefully, such problems will be solved by time.

A reason for the current situation may be that BPSim is a new standard and there currently only a few competitors on the market.

However, the BPSim specification has also some structural problems that will be discussed in the following sections.

Use of Expression Parameters has Limits

For many scenarios, adding parameters to BPMN elements by means of the BPSim extension works well. For more complicated cases, BPSim allows to add properties to a process instance as well as to BPMN elements.

For example, in our reporting process, a property of a process instance could be the number of pages of the report. Such process instance properties can be regarded as global variables that can be read and written in the context of each BPMN element. If the upper path in the diagram of Fig. 1 is taken, a simple report has to be compiled (number of pages = 30) while otherwise a long report has to be compiled (number of pages = 100). When the decision has been taken, the property “numberOfPages” is set to the appropriate number. Using so-called expression parameters, it is also possible to define that the costs for the task “print report” depends on the number of pages (say 2 cents per page). This way, by reading and modifying properties, some additional logic (that cannot be seen in the BPMN diagram) can be added to the model.

```
<bpsim:CostParameters>
...<bpsim:fixedCost>
…...<bpsim:ExpressionParameter value=
        "bpsim:getProperty('numberOfPages') *0.02"/>
  </bpsim:fixedCost>
</bpsim:CostParameters>
```

Figure 8

In this case, BPSim specifies that the parameter bpsim:CostParameters (Fig. 8) is serialized as an XML *element*, and the content of this element can be an expression parameter.

The situation is different if we try to model the processing time of the print task in the same manner. We can specify that the time is given by a truncated normal distribution with a mean of 70 (and a minimum and maximum value) as follows (Fig. 9):

```
<bpsim:TimeParameters>
  <bpsim:ProcessingTime>
    <bpsim:TruncatedNormalDistribution
        max="1000" mean="70" min="0"
       standardDeviation="10"/>
  </bpsim:ProcessingTime>
</bpsim:TimeParameters>
```

Figure 9

In Fig. 9, the distribution parameters such as “mean” are *attributes* (in this case with the data type Double) in the XML serialization, and the standard provides no means to express them as a calculated value (i.e. as an BPSim expression parameter) as it was the case for the costs.

BPSim Semantics is Interrelated with BPMN Semantics

Let’s assume that we want to interrupt the task “print report” if it took more than 5 minutes. This can be expressed in plain BPMN: A boundary timer event is added to the task (see Fig 2), and it is provided with a *TimerEventDefinition* attribute that specifies that the event fires 5 minutes after the task has been started (Fig. 10):

```
<boundaryEvent id="cancelPrintTimer_id"
          name="Cancel Print" cancelActivity="true"
          attachedToRef="PrintTaskID">
  <timerEventDefinition>
     <timeDuration>PT5M</timeDuration>
  </timerEventDefinition>
</boundaryEvent>
```

Figure 10

In this case, the timing behavior is completely defined in BPMN (not using the BPSim extension), and BPSim does not provide a standard way to say “interrupt the task if it took more than 5 seconds multiplied with the current value of the “number of pages” attribute. Although the timing behavior of the boundary event could be defined using BPSim as well, this would not be appropriate because the BPSim attribute *InterTrigger Timer* that would have to be used for this purpose cannot be related to the point of time when the task “print report” has been started. What would be needed, but is not included in the standard, is the possibility to deal with different timers which can be reset when a task starts (or in general: when an event occurs).

Resource Model not yet Fully Elaborated

Next, let’s assume that before starting to print, the printer needs a warm-up period of 3 minutes if the last print job ended more than 20 minutes ago. For modeling such a situation, the possibility to reset a timer (this time when a task ends) would be required again. In addition, it would be useful if the printer (a resource) would have a time parameter denoting the needed warm-up time as well as a property parameter for storing the information when the last print job ended. Unfortunately, according to BPSim, both kinds of parameters are not allowed for resources.

Altogether, BPSim uses an advanced, but not yet fully elaborated resource model. Resources can have more than one role. Priorities can be assigned to activities. Also, the availability of resources can be defined depending on time intervals (e.g. representing shifts). It is possible to model the (un)availability of resources as a random variable in order to deal with illness or malfunction of technical resources. However, there are still things missing. Although activities can have priorities, the standard does not say anything about the semantics of such a priority attribute. It can be assumed that the meaning of priorities is that a resource when becoming available is assigned to the activity with the highest priority (a feature that [Wal06] requires for useful business process simulation) – but it should be possible to define other resource allocation strategies as well. Even if two activities have no priority information (or both have the same), it should be possible to specify whether a resource is allocated to a random activity, to the most recent one (LIFO), to the one which has been waiting for the longest time (FIFO), etc.

There is no means for modeling consumable resources (such as raw material) that will not be released when a task that needs a resource is completed. While such information could be modeled as a property of a process instance, such a way of modeling is less intuitive than having a richer resource model. In particular, resources should be allowed to have user-defined property parameters (which is currently not the case).

A richer resource model would also be very useful for modeling working preferences, locations and working speed of resources. In [Wil M. P. van der Aalst et. al 2009], it is discussed that current simulation tools often use oversimplified resource models. Among others, it is not taken into account that people do not work on a constant speed and tend to work part times or in batches. Other than assuming that a resource is available as soon as it is required by a task, simulation models should be able to support various resource patterns [Nick Russel et. al. 2005]. While the support of such rich resource models has been announced as one of the goals of the BPSim initiative [Jan11], the resource model in version 1.0 of the standard has still room for improvement.

Working with historical process data sets

Often, simulation models have a lot of parameters such as duration times, interarrival times and probabilities for decisions. Accordingly, a lot of replications are required to get statistical valid interpretations of the simulation.

In a typical process improvement projects, the data of historical process instances are known from the logs of BPM engines. In order to build a realistic simulation model, it makes sense to use randomly generated values only for those parts of the model, for which no historical data are available. This approach reduces the number of randomly generated parameters, and the number of required replications can be reduced considerably.

BPSim supports working with historical datasets. In a BPSim model, these datasets are assigned to simulation parameters such as decisions or duration times of tasks . However a weakness of the approach is that this assignment is always done in the context of the whole process and does not refer to process instances. In our example (Fig. 1), this would mean that historical data can be used for simulating the decisions and the duration of the tasks in the process. However, the fact that the task “print report” takes longer when the decision “long report required” has been taken, would not be considered in the model.

Result Types are Insufficient

A weakness of the BPSim standard is that the result types that can be requested from a simulation are too limited. Allowed result types are the number of occurrences, minimum, maximum and mean values. However, average, best and worst case scenarios (represented by the minimum and maximum values) are often not enough to describe the statistical distribution of the simulation results. At least, an information about standard deviation and skewness is desirable. In addition, we have to ask for percentiles as well if we want to deal with service-level agreements such as “95% of the requests have to be handled within n time units”. Unfortunately, such descriptors have not been considered in the BPSim standard. In general, it would be desirable to require that a simulation tool should write a log (in a standardized format) containing all events and decisions happening during a simulation run. This would allow any analysis after a simulation run.

CONCLUSIONS

The motivation behind the BPSim specification was to close the gap between the well-established BPMN standard for modeling and a great variety of simulation tools, each one requiring a proprietary input format. Having such a standard can help to promote the use of business process simulation and to build tools for modeling simulation models independently from simulation tools.

BPMN and BPSim are powerful enough to create models for business process which are „static“ in the sense that parameters may be random variables, but the distribution of those random variables does not change during the process. However, we see from the above examples that the BPSim standard needs improvement for cases where probability distributions change when the process is executed.

Also, it has to be noted that neither BPMN nor BPSim has a fully elaborated resource model. For simulation purposes, a more detailed metamodel for resources (a suggestion can be found in [Cristina Cabanillas 2011]) would be desirable.

Additionly, the semantics of BPMN and its extension BPSim can lead to contradictions. Also, historical data and result types do not support the im- and export of raw data. At this point BPSim shoud extended.

Neither in the investigation for [Christian Müller et. al. 2015a and 2015b] nor in the preparation on this paper we found tools that have a full BPSim support. All current tools support the standard partially and have additional vendor extensions. In one case it was necessary to modify a model generated by a BPSim modeler with a text editor for running it in a simulation engine. These examples show that, in contradiction to the BPMN environment, the interchangeability of models between tools is not yet satisfactory. The authors hope that this will be changed by time.

REFERENCES

Wil M. P. van der Aalst; Joyce Nakatumba; Anne Rozinat and Nick Russell 2009: Business Process Simulation: How to get it right? International Handbook on Business Process Management, Springer, 2009

BPMN 2013: ISO/IEC International Standard 19510: Information Technology – Object Management Group Business Process Model and Notation, Document Number ISO/IEC 19510:2013(E), 2013

BPSim 2013: Workflow Management Coalition: BPSim – Business Process Simulation Specification, Document Number WFMC -BPSWG-2012-1, 2013

BPSim 2014: Workflow Management Coalition: BPSim Implementer's Guide, 2014

Cristina Cabanillas, Manuel Resinas, Antonio Ruiz-Cortés 2011: RAL: A High-Level User-Oriented Resource Assignment Language for Business Processes, BPM 2011 Workshops, LNBIP Vol 99, Springer, 2011

Nick Russell, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, David Edmond 2005: Workflow Resource Patterns: Identification, Representation and Tool Support. CAiSE 2005: 216-232

John Januszczak, Geoff Hook 2011: Simulation standard for business process management. Winter Simulation Conference 2011: 741-751

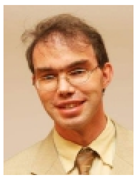
Christian Müller et al. 2015A: Gegenüberstellung der Simulationsfunktionalitäten von Werkzeugen zur Geschäftsprozessmodellierung, TH Wildau, http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:kobv:526-opus4-4354

Christian Müller, Klaus Bösing 2015b: Vergleich von Simulationsfunktionalitäten in Werkzeugen zur Modellierung von Geschäftsprozessen, in AKWI 2015, http://dx.doi.org/10.15771/978-3-944330-47-1_2015_1

Raimar Scherer 2011: Process-Based Simulation Library for Construction Project Planning, Winter Simulation Conference 2011

Anthony Waller, Martin Clark, Les Enstone 2006: L-SIM: Simulating BPMN Diagrams with a Purpose Built Engine, Winter Simulation Conference 2006

AUTHORS BIOGRAPHIES



RALF LAUE studied mathematics at the University of Leipzig, Germany. After graduating, he worked as a system programmer before returning to the University of Leipzig in 2003. He obtained a PhD in computer science in 2010. Since 2011, he is a full professor for software engineering at the University of Applied Sciences in Zwickau, Germany. His research interests include the correctness and understandability of visual models in computer science.

His email address is: ralf.laue@fh-zwickau.de

CHRISTIAN MÜLLER has studied mathematics at Free University Berlin. He obtained his PhD in 1989 about network flows with side constraints. From 1990 until 1992 he worked for Schering AG and from 1992 until 1994 for Berlin Public Transport (BVG) in the area of timetable and service schedule optimization. In 1994 he got his professorship for IT Services at Technical University of Applied Sciences Wildau, Germany. His research topics are conception of information systems plus mathematical optimization and simulation of business processes.

His email address is: christian.mueller@th-wildau.de and his web page is http://www.th-wildau.de/cmueller/ .