Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

SIMULATION-BASED PERFORMANCE AND RELIABILITY ANALYSIS OF BUSINESS PROCESSES

Paolo Bocciarelli
Andrea D'Ambrogio
Andrea Giglio
Emiliano Paglia

Department of Enterprise Engineering
University of Rome Tor Vergata
Via del Politecnico 1
00133, Rome, ITALY

ABSTRACT

The use of process modeling combined with the use of simulation-based analysis provides a valuable way to analyze business processes (BPs) and to evaluate design alternatives before committing resources and effort. The simulation-based analysis of BPs usually addresses performance in terms of efficiency, i.e., focusing on time-related properties (e.g., throughput or execution time). Differently, this paper proposes an automated method for the analysis of BPs in terms of both efficiency-related performance and reliability. In addition, the method allows business analysts to carry out a joint performance and reliability analysis by introducing a so-called performability attribute. The proposed method is illustrated by use of a running example dealing with a conventional e-commerce scenario.

1 INTRODUCTION

Over the last decades, simulation has proven to be a valuable technique to enact an effective and timely analysis of complex systems. Simulation-based approaches give the capability to predict the behavior of systems from the early stages of the development lifecycle and enable designers and stakeholders to assess whether or not such systems accomplish both functional and non-functional requirements.

The adoption of simulation-based techniques may constitute a winning move in the Business Process Management (BPM) domain, in which the competitive and dynamic nature of the global marketplace pushes enterprises to enact a continuous effort aimed at the improvement of provided services and goods. Enterprises need to rapidly meet market changes in order to gain and maintain a prominent position. In this respect, the use of business process modeling combined with the adoption of simulation-based analysis provides a cost effective, accurate, and rapid way to evaluate alternatives before committing the required effort and resources (Tumay 1996, van der Aalst et al. 2010).

On the other hand, the concrete use of simulation-based analysis of Business Processes (BPs) is still limited, mainly due to the fact that the specification, implementation and execution of simulation models require a non-negligible effort and significant skills (Hook 2011; Kamrani, Ayani, and Karimson 2010; van der Aalst et al. 2010).

Existing BPM engines and tools, e.g., jBPM (JBoss Community 2014), Bizagi BPM suite (Bizagi 2014), etc., include specific capabilities for simulating BP models, yet with a limited effectiveness. Indeed, very often such tools exhibit a low degree of customization and/or provide a rough representation of the real BP behavior, as the BP simulation merely consists of a simple model animation.

978-1-4799-7486-3/14/$31.00 ©2014 IEEE

In this context, this paper proposes a method to automate the simulation-based analysis of BPs. The proposed method adopts model-driven standard and tools and exploits eBPMN (Bocciarelli et al. 2014), a domain-specific simulation language based on the execution semantics of BPMN (Business Process Model and Notation), the standard language for BP specification (OMG 2011).

The simulation-based analysis of BPs usually focuses on the performance behavior of processes from the efficiency point of view only (e.g., in terms of throughput or execution time), without taking into account the important issue of process reliability, i.e., the probability that the BP performs correctly in a given timeframe (often referred to as *mission time*). The ability to predict the reliability of a BP is instead important to assess the effectiveness of a performance improvement or optimisation. Reliability is indeed directly associated with the overall process quality and in some cases reliability and efficiency may provide a conflicting contribution, i.e., improving reliability may lead to an efficiency decrease and vice versa, as shown later on.

In order to take into account the combined effects of efficiency and reliability to the overall process quality, this paper focuses on the so-called *performability* attribute, i.e., a joint measure of efficiency and reliability (Smith and Kishor 1988).

From the reliability point of view, BPMN natively provides constructs to represent issues that affect or alter the BP execution flow (e.g., operation failures, error events, timeout events, etc.). Furthermore, BPMN provides mechanisms to specify actions that have to be carried out to ensure the consistency of the failed BP instance (e.g., compensations, errors handling, etc.). All such failure- and error-related events are specifically considered by the BP designer and explicitly introduced in the BPMN model. Failures of such a kind are similar to exception handlers, which are commonly included into software application to properly handle expected computational issues.

Differently, in the proposed approach we aim to introduce a *reliability analysis* that takes into consideration *unexpected failures* or, in other words, those events that are not natively specified in standard BPMN models and that cause the abnormal interruption of the affected process instance (e.g., a power outage that forces a server switch off or causes an hardware damage, a software bug that causes an unrecoverable failure, etc.).

The proposed method is based on an hybrid approach that combines analytical and simulation-based techniques. More specifically, the method provides an analytical approach based on a graph reduction algorithm to predict the reliability of BPs, while the performance-related prediction is obtained by use of eBPMN. Finally, the method introduces an algorithm that combines the performance and reliability predictions into a performability prediction.

The predictions are obtained by use of an automated model-driven method that takes as initial input an annotated BPMN model of the BP under study. The annotated BPMN model includes a performance and reliability characterization of the BP. To this purpose, this work makes use of an extended version of PyBPMN (Performability-enabled BPMN), a lightweight BPMN extension that addresses the specification of performance and reliability properties (Bocciarelli and D'Ambrogio 2011).

The remainder of this work is structured as follows: Section 2 reviews related contributions, Section 3 illustrates the performability-oriented characterization of BPs, while Section 4 describes the model-driven method for the performability prediction, by use of a running example application dealing with an e-commerce scenario. Finally, Section 5 gives the concluding remarks.

2 RELATED WORK

Various contributions can be found in literature that deal with the reliability analysis of BPs. In van der Aalst (2010), van der Aalst et al. (2010) the authors focus on the process availability, i.e., the percentage of time the process is available to be executed. The authors show the most frequent pitfalls in modeling resources, mainly related to the fact that resources may be involved in multiple processes and tend to work in batches (in case of human resources). This facts can have dramatic effects on the key performance indicators of a process.

A deep analysis of the state of the art of BP performance analysis is illustrated in (van der Aalst 2013). The author identifies three dimensions of performance, i.e., time, cost and quality, but does not give any information about the reliability dimension.

In Vanderfeesten et al. (2007) the authors elaborate on the importance of quality metrics for business process modeling, demonstrating that the design of software shares many similarities with the design of BPs.

In Brall (2010) the author gives a precise definition of reliability for business processes and states that the use of reliability tools is a cost effective approach to prevent business process failures. An alternative modeling and optimization approach dealing with BP reliability is presented in (Lam et al. 2010). The proposed method uses a communication network of probabilistic graphs and further analyzes and simulates the business process network model using system dynamics.

All such contributions address either performance or reliability attributes but do not combine them into a joint attribute. Differently, this paper focuses on the performability attribute and also introduces a method to obtain a performability prediction of a BP specified in BPMN.

Finally, it should be underlined that this paper is founded on and extends previous contributions. Specifically, the PyBPMN metamodel adopted in this paper extends the one introduced in Bocciarelli and D'Ambrogio (2011), to allow the representation of different concrete configurations implementing the same abstract BP. Moreover, the eBPMN language, which has been presented in Bocciarelli et al. (2014), is adopted in this work to carry-out the simulation-based performance analysis of BPs. Finally, the model-driven method and the related model transformations at the basis of this work are an extension of previous contributions (Bocciarelli and D'Ambrogio 2014, Bocciarelli et al. 2014), which have been here refined and improved to effectively benefit from the eBPMN language and the revised PyBPMN metamodel.

3 PERFORMABILITY-ORIENTED CHARACTERIZATION OF BPS

The BPMN language does not natively provide any construct to associate performance or reliability properties to process elements.

In order to specify a performability-oriented characterization of BPs, this work introduces the use of *text annotations*, which provide the required information according to a syntax based on PyBPMN.

A BP is a collection of interconnected *activities*, which are executable elements that can be atomic (i.e., *tasks*) or non atomic (i.e., sub-processes). The execution of a process task requires the availability of specific *resources*, i.e., human resources, devices and/or software services.

A BPMN model provides an abstract description of a BP in terms of a set of tasks. The BP implementation is then obtained through the allocation of concrete resources to tasks. A given BPMN model can thus be mapped to several different BP implementations, in case several alternative resources are available for executing one or more tasks.

The allocation of resources to tasks leads to the identification of several *candidate configurations*, among which selecting the one for the actual BP implementation. For instance, a given *service task* could be implemented by different web services, as well as a given *manual task* could be completed by different persons. A given BP implementation makes use of a specific web service and a specific person for the service task and the manual task, respectively.

According to this perspective, the BPMN model must be appropriately annotated to specify both the resource allocation and the performance and reliability characterization of such resources. Such annotation is carried out by use of PyBPMN, as described in the next subsection.

3.1 PyBPMN-Based Annotation of BPMN Models

PyBPMN (Performability-enabled BPMN) is a lightweight BPMN extension that addresses the specification of performance and reliability properties of a BP (Bocciarelli and D'Ambrogio 2011). In order to enable the representation of different concrete configurations implementing the same abstract BP, this work provides

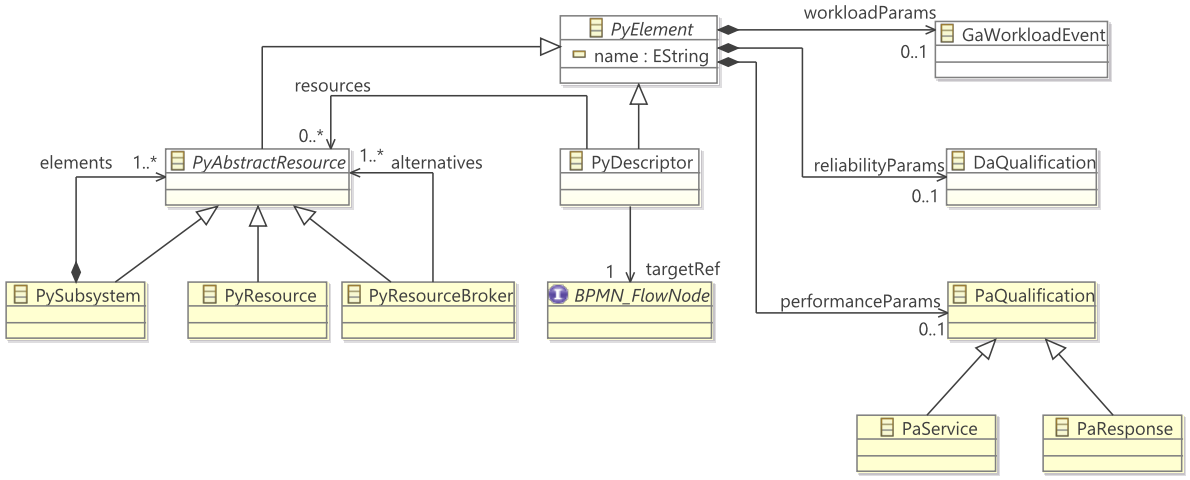

Figure 1: Key metaclasses of the revised PyBPMN metamodel.

an extension of the PyBPMN metamodel. More specifically, the revised PyBPMN model allows business analysts to represent the set of *resources* associated to each BPMN `FlowNode`[1]. Each resource is characterized by different performance and reliability properties.

Figure 1 shows the key metaclasses of the revised PyBPMN metamodel. The `PyElement` is the base abstract metaclass, used to specify workload (`GaWorkloadEvent`), reliability (`DaQualification`) and performance (`PaQualification`) properties[2]. Such properties can be associated either to `FlowNode` elements (as in the original definition of PyBPMN) through a `PyDescriptor` element or to resources being referenced by the `PyDescriptor` element.

The proposed resource modeling is flexible enough to enable a fine grained specification of resource usage. Resources are modeled using the composite pattern, thus enabling the specification of either a simple resource (`PyResource`) or a subsystem (`PySubsystem`), which is composed by any set of simple resources and other subsystems.

The representation of different concrete configurations for a BP is obtained associating alternative resources to the same `FlowNode` element, denoted as `BPMN_FlowNode` in Figure 1 to specify that it is imported from the BPMN metamodel.

The `PyResourceBroker` metaclass has been introduced to enforce the selection of a resource over a set of alternative resources (which can be simple resources or subsystems). For example, if a `FlowNode` element is allocated to two kinds of resources, where one kind (resource A) is known and the other kind can be alternatively implemented by two concrete resources (resources B1 and B2), the `PyDescriptor` element associated to the `FlowNode` element will have two resource references: one to `PyResource` A and one to a `PyResourceBroker` element having `PyResource` B1 and `PyResource` B2 as `alternatives` association ends.

The definition of resources and their performability parameters is carried out in the BPMN language using `TextAnnotation` elements with a specific syntax, according to the following general form, where VSL stands for Value Specification Language (OMG 2009):

```
<<meta-class name>>{VSL expression}
```

[1]A *Flow Node* element is used in the BPMN metamodel to provide the single source and target elements of a sequence flow that shows the order of elements in a process. The `Activity` metaclass, which is used to represent the work tasks of a process, is a sub-class of the `FlowNode` metaclass.

[2]The reader is sent to (Bocciarelli and D'Ambrogio 2011) for a detailed description of the relevant attributes.

Such a solution, based on standard BPMN elements such as `TextAnnotation` elements, allows business analysts to specify resource parameters using any BPMN editor. As an example, the following `TextAnnotation` element specifies a resource with name `resource1`, service time of 250 milliseconds and MTTF (mean time to failure) of 50000 hours :

```
<<PyResource>>{
  name=resource1,
  performanceParams=(<<PaService>>{serviceTime=(value=250, unit=ms)}),
  reliabilityParams=(<<DaQualification>>{MTTF=(value=50000, unit=hours)})
}
```

A `PyResourceBroker` element is instead used to specify alternative configurations. As an example, the following `TextAnnotation` element specifies a resource broker for two alternative resources (`resource1` and `resource2`):

```
<<PyResourceBroker>>{alternatives={resource1, resource2}}
```

The resources allocated to a `FlowNode` element are then specified associating the corresponding `TextAnnotation` elements.

4 MODEL-DRIVEN METHOD FOR PERFORMABILITY PREDICTION

This section illustrates the model-driven method for the performability prediction of BPs. The method provides an effective support along the whole BP development lifecycle, from the BP specification down to the BP execution.

The rationale of the method is summarized in Figure 2. In the next subsections each step is further illustrated by use of a running example application dealing with an e-commerce scenario referred to as *purchase service*.

4.1 Business Process Specification

The business process is first specified in terms of a BPMN model, which a business analyst specifies according to the BP functional requirements.

The aforementioned *purchase service* example application includes the following main steps:

1. the customer selects the *checkout* button to complete the order of given items;
2. the customer specifies the information required to pay (e.g., the credit card number, the shipping address, etc.) and complete the payment;
3. the invoice is created and forwarded to the customer;
4. the purchased items are packaged and shipped to the customer.

As regards step 2, it is assumed that the payment fails with a 10% probability. In this case the process terminates.

The following tasks are required to specify the *purchase service* business process:

- a task for managing the checkout procedure, denoted as *CheckOut (CO)* task;
- a service task for managing the payment procedure, denoted as *PaymentManager (PM)* task;
- a service task for managing the billing procedure, denoted as *BillingManager (BM)* task;
- a service task for managing the shipment procedure, denoted as *ShipmentManager (SM)* task.

4.2 BPMN Annotation

The abstract BPMN model specified at the previous step is enriched with textual annotations describing the allocation of resources to tasks, along with the performance and reliability properties of such resources.

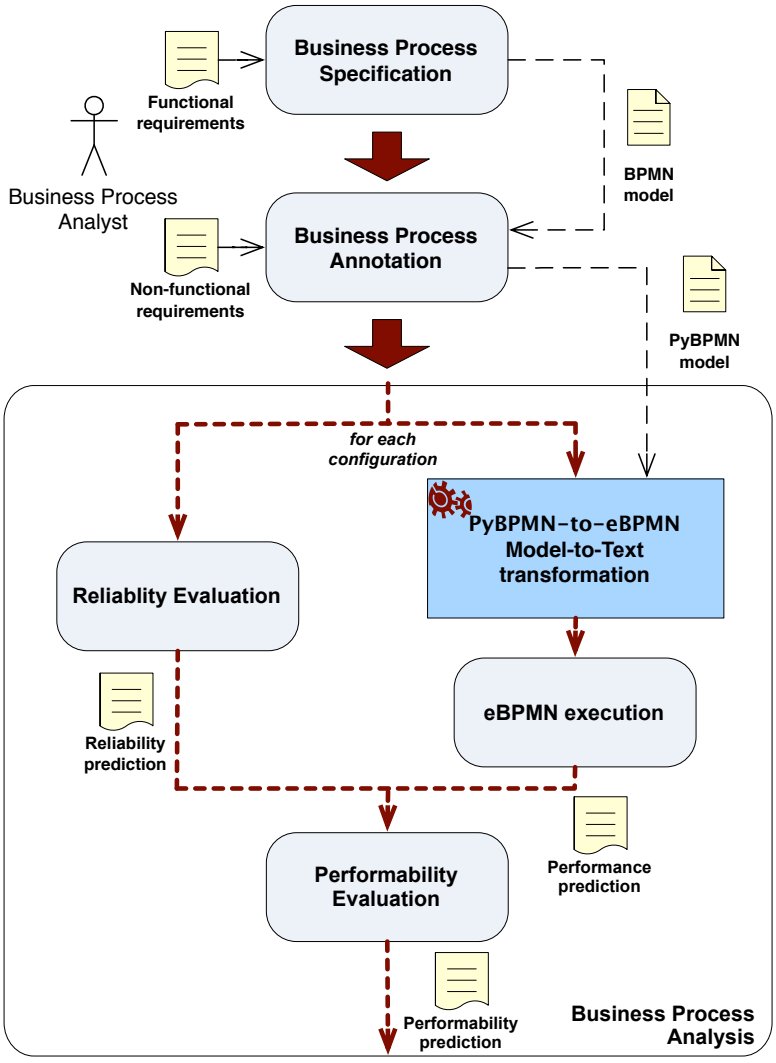

Figure 2: Method for performability analysis of BPs.

The BPMN annotation step is carried out according to the syntax specified in Section 3.1. The obtained model is referred to as *PyBPMN model* in Figure 2.

As stated in Section 3, the association of alternative resources to the set of BP tasks leads to the definition of various candidate configurations.

Let us now assume that two different resources are available for implementing the *PM* task of the example application. Table 1 summarizes the performance and reliability attributes of the two alternative configurations, namely PM_A and PM_B, obtained by associating the two available resources to the *PM* task. The values for the reliability R parameter are obtained from the relevant MTTF, assuming the following exponential failure distribution and a mission time t of one year:

$$R(t) = e^{-\frac{1}{MTTF}t} \tag{1}$$

The resulting PyBPMN model, which includes the resource allocation with the performance and reliability properties for each task, is illustrated in Figure 3.

Table 1: Performance and reliability parameters of configurations PM_A and PM_B.

Parameter		PM_A	PM_B
Performance	PaymentManager service time	310 ms	280 ms
Reliability	MTTF (mean time to failure)	$5.00*10^8$ s	$1.49*10^8$ s
	R(1year)	0.939	0.809

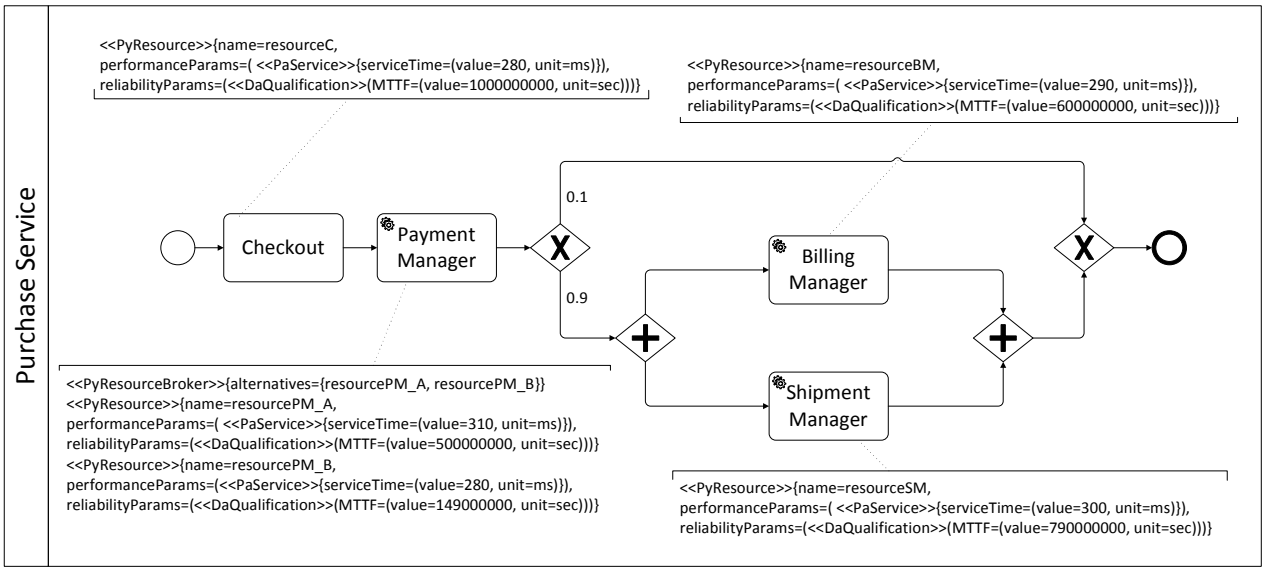

Figure 3: PyBPMN model for the example application.

Business analysts and designers may thus be interested to compare the different alternatives in order to identify the configuration that provides the best performance and reliability and/or to answer what-if questions.

4.3 Business Process Analysis

The business process analysis step is the core of the proposed method. The step is carried out by first obtaining the performance and reliability predictions for each candidate configuration and then combining them into a performability prediction that allows to select the optimal candidate configuration, which is then used for BP implementation and execution.

In the example application case, the performance and reliability predictions, as well as the performability prediction, are to be obtained for the PM_A and PM_B configurations.

The next subsections give the details of the BP analysis step.

4.3.1 Performance Analysis

The performance analysis is carried out by use of eBPMN, a domain specific simulation language built according to the BPMN 2.0 execution semantics (Bocciarelli et al. 2014).

As depicted in Figure 2, the proposed method does not require an explicit use of eBPMN (and the knowledge of its syntax and architecture), as the eBPMN code is automatically generated.

Specifically, the performance analysis is carried out for each candidate configuration throughout the two following steps: i) the `PyBPMN-to-eBPMN` *model-to-text* transformation is executed to automatically obtain the eBPMN code starting from the PyBPMN-based specification of the BP under study; ii) the eBPMN code is then executed to obtain the performance prediction.

The `PyBPMN-to-eBPMN` transformation is specified and implemented in Acceleo (Eclipse Foundation 2014), the standard language for specifying model-to-text transformations, provided by OMG as part of the MDA effort (OMG 2003). Details about the `PyBPMN-to-eBPMN` transformation can be found in Bocciarelli et al. (2014).

The automated model transformation makes the method easy to use also for business analysts who are not familiar with simulation languages and tools, since they are only required to provide the PyBPMN model resulting from the BPMN annotation step of the method.

Figure 4 summarizes the results of the performance prediction for the example case study, in terms of throughput (i.e., completed process instances per second) of the BP with resources PM_A and PM_B, for different values of the process execution requests in terms of interarrival rate.

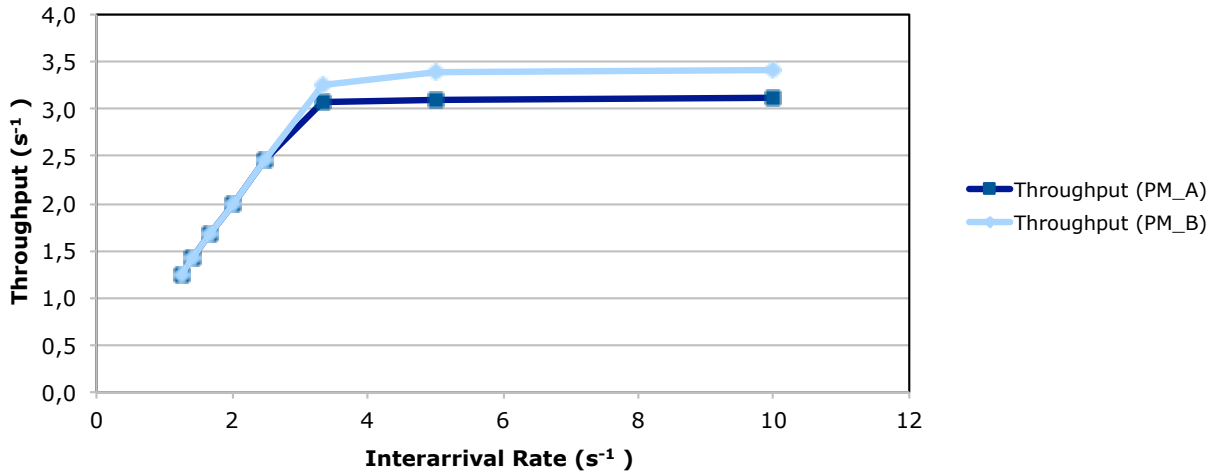

Figure 4: Performance prediction for the example application.

A validation of the performance prediction obtained by use of the eBPMN language has been carried out by comparing the outcomes with real data monitored on a set of test service-oriented business processes, as discussed in Bocciarelli and D'Ambrogio (2008), Bocciarelli and D'Ambrogio (2011).

According to the performance prediction, the configuration including PM_B is the one to be preferred.

4.3.2 Reliability Analysis

The reliability analysis takes as input the PyBPMN model of the BP under study and yields as output the reliability prediction for each configuration. The prediction is carried out by use of an algorithm that iteratively applies a set of reduction rules until only a single atomic *flow node* remains. The algorithm, inspired by (Cardoso et al. 2004), applies the reduction rules shown in Figure 5.

The structure of the annotated PyBPMN changes at each iteration and after a number of iterations it is reduced to a single node. The reliability associated to the so obtained node specifies the reliability of the whole process.

The reliability analysis applied to the example case study leads to the following reliability predictions (over a mission time of one year): $R_{PM_A}(t) = 0,838$, $R_{PM_B}(t) = 0,722$.

In this case the configuration including PM_A is the one to be preferred, differently from what stated according to the performance prediction illustrated in Section 4.3.1. As the performance and the reliability predictions lead to conflicting results, a performability prediction is to be used to take a decision about the configuration to be selected.

4.3.3 Performability Analysis

Let us consider n different configurations BP_i $(i = 1..n)$ of a BP. Each configuration may be analyzed in terms of performance and reliability by use of the methods illustrated in the previous sections, in order to obtain the prediction that leads to an optimal choice of the initial configuration in terms of either performance or reliability.

At this time, it may happen that conflicting predictions are found, i.e., the optimal configuration selected in terms of performance is not the optimal one in terms of reliability and vice versa.

This claims for a joint analysis of performance and reliability, in other words for a performability analysis that is carried out by use of the following algorithm (Bocciarelli and D'Ambrogio 2014):

1. generate a *state transition diagram (STD)* in which states represent the possible configurations that the BP implementation may undergo and edges represent the transitions from a given configuration to a different one (this implies that when a resource fails and an alternative working resource for the same task is available, the business process implementation switches to a new configuration that includes the working resource);
2. select a candidate configuration as the initial configuration;
3. use the reliability prediction method illustrated in Section 4.3.2 to obtain the transition probabilities of the STD;
4. calculate the absorbing probabilities $P(BP_i)$ of being in a given working configuration $i = 1..n$ starting from the initial configuration;
5. use the performance prediction method illustrated in Section 4.3.1 to obtain the performance associated to each configuration, e.g., in terms of its throughput $T(BP_i)$, and assign it as a reward to the configuration;
6. obtain the performability prediction in terms of the expected reward rate of BP as follows:

$$RW(BP) = \sum_{i=1}^{n} P(BP_i)T(BP_i) \tag{2}$$

where:

- $RW(BP)$ is the expected reward rate of the business process, i.e., an overall attribute that combines performance and reliability;

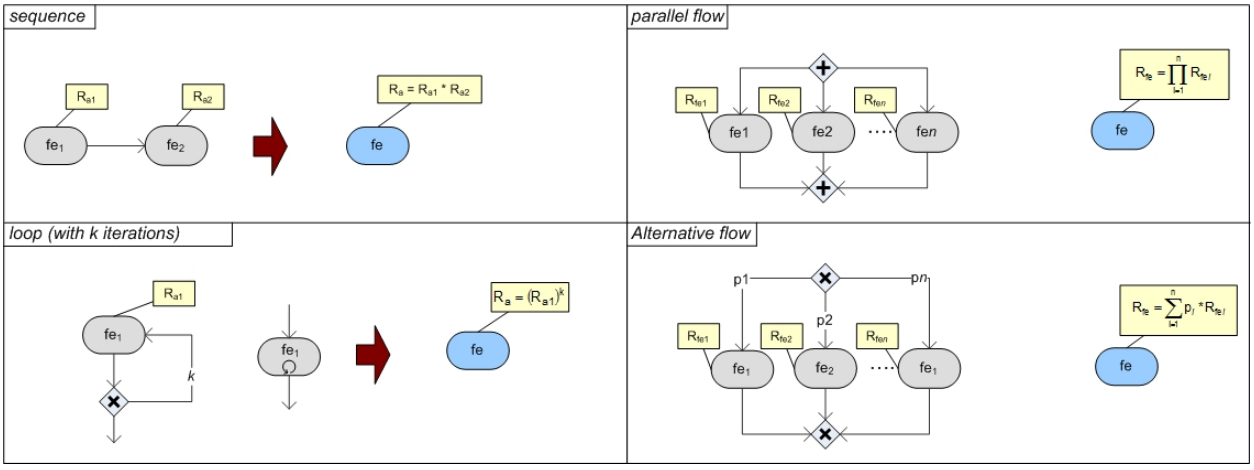

Figure 5: Reduction rules for computing the reliability of annotated BPs.

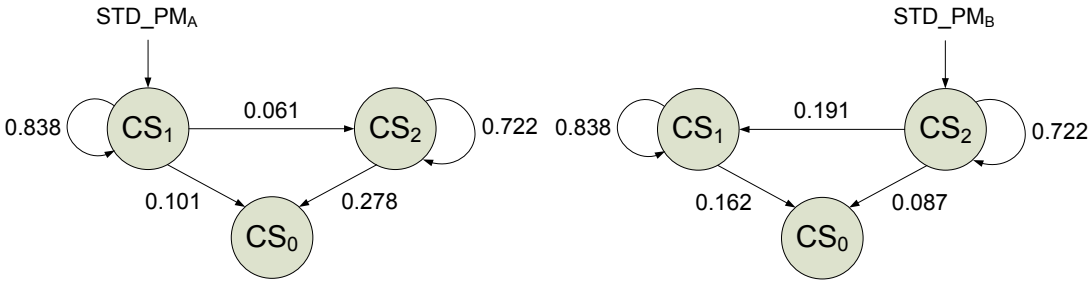

Figure 6: STD for the example application (alternatives STD_PM_A and STD_PM_B).

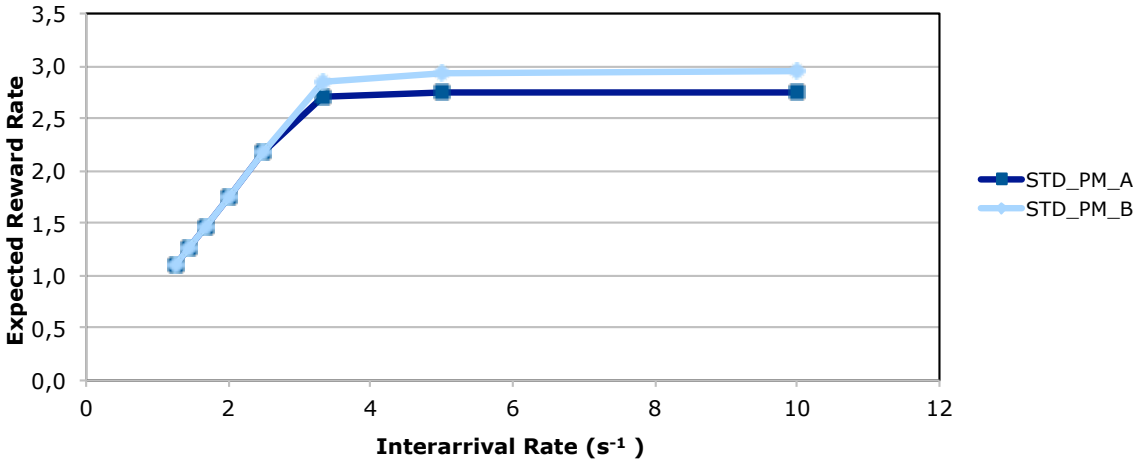

Figure 7: Expected reward rate for the example application.

- $P(BP_i)$ is the probability of the system to be in the $i-th$ working configuration starting from the initial configuration, as computed by means of the STD;
- $T(BP_i)$ is the throughput of the $i-th$ candidate configuration.

The comparison among the so obtained reward rates for each candidate initial configuration allows one to carry out a choice that takes into account both the performance and the reliability of the business process.

Figure 6 gives the STD for the example case study. The state CS_1 represents the business process in the configuration that includes PM_A, while the state CS_2 represents the business process in the configuration that includes PM_B. The state CS_0 represents the business process in the failed state.

According to the STD, two different alternatives may be considered for the initial configuration: the first one (denoted as STD_PM_A) assumes CS_1 as the initial configuration and CS_2 as a backup configuration in case of PM_A failure, while the second one (denoted as STD_PM_B) assumes CS_2 as the initial configuration and CS_1 as a backup configuration in case of PM_B failure.

The performability prediction of the example business process is given in Figure 7, which depicts the expected reward rate for the BP with PM_A and PM_B.

Figure 7 shows that an initial configuration with PM_B is to be preferred, in contrast with what obtained from the reliability prediction and according to the performance prediction.

The simple but effective case discussed herein gives an example of the importance of combining the analysis of performance and reliability.

5 CONCLUSIONS

Simulation is an effective technique for analyzing BPs and a valuable tool for business analysts dealing with the continuous improvement of enterprise processes. Usually, simulation-based approaches focus on the analysis of the BP performance in terms of efficiency attributes only, without taking into account the important issue of process reliability.

This paper has introduced a method for the BP analysis in terms of performability, a joint measure of performance and reliability. The proposed method makes use of a model transformation approach that provides a significant degree of automation. This enables business analysts to easily get the performability prediction without being required to be familiar with specific simulation languages and with performance and reliability theory.

The method has been applied to a simple but effective example that demonstrates the importance of carrying out a performability analysis when separate performance and reliability predictions lead to conflicting results.

The proposed method does not take into account the monetary loss value associated to failures, in other words, the fact that a BP having a significant failure probability with a negligible cost is not as serious as a BP having a reduced failure probability with remarkably higher cost. Work is in progress to address the loss value by appropriately customizing the BP annotations and revising the prediction method.

REFERENCES

Bizagi 2014. "Bizagi BPM Suite". Available at http://www.bizagi.com/.

Bocciarelli, P., and A. D'Ambrogio. 2008. "Model-driven Performability Analysis of Composite Web Services". In *Proceedings of the 2008 SPEC Intl. Performance Evaluation Workshop*, 228–246: Springer.

Bocciarelli, P., and A. D'Ambrogio. 2011. "A BPMN Extension for Modeling Non Functional Properties of Business Processes". In *Proceedings of the 2011 Symposium on Theory of Modeling and Simulation*, 160–168: Society for Computer Simulation International.

Bocciarelli, P., and A. D'Ambrogio. 2014. "A Model-driven Method for Enacting the Design-time QoS Analysis of Business Processes". *Software & Systems Modeling* 13:573–598.

Bocciarelli, P., A. D'Ambrogio, A. Giglio, E. Paglia, and D. Gianni. 2014. "Empowering Business Process Simulation Through Automated Model Transformations". In *Proceedings of the 2014 Symposium On Theory of Modeling and Simulation*: Society for Computer Simulation International.

Bocciarelli, P., A. D'Ambrogio, and E. Paglia. 2014. "A Language for Enabling Model-driven Analysis of Business Processes". In *Proceedings of the 2nd International Conference on Model-Driven Engineering and Software Development*: SciTePress.

Brall, A. 2010. "Reliability analysis - a tool set for improving business processes". In *Proceedings of the 2010 Reliability and Maintainability Symposium*. Institute of Electrical and Electronics Engineers, Inc.

Cardoso, J., J. Miller, A. Sheth, J. Arnold, and K. Krys. 2004. "Quality of service for workflows and web service processes". *Journal of Web Semantics* 1:281–308.

Eclipse Foundation 2014. "Acceleo". Available at http://www.acceleo.org/pages/home/en.

Hook, G. 2011. "Business Process Modeling and simulation". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. C. Fu, 773–778. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

JBoss Community 2014. "JBoss Business Process Management". http://www.jboss.org/jbpm.

Kamrani, F., R. Ayani, and A. Karimson. 2010. "Optimizing a Business Process Model by Using Simulation". In *Proceedings of the 2010 IEEE Workshop on Principles of Advanced and Distributed Simulation*, 1–8. Institute of Electrical and Electronics Engineers, Inc.

Lam, C. Y., S. L. Chan, and W. H. Ip. 2010. "A Structural Reliability Business Process Modelling with System Dynamics Simulation". In *Modeling Simulation and Optimization–Focus on Applications*, edited by S. Cakaj, Chapter 16, 259–267. Rijeka, Croatia: InTech.

OMG 2003. *MDA Guide, version 1.0.1.*

OMG 2009. *A UML profile for Modeling and Analysis of Real Time Embedded Systems, v. 1.0.*

OMG 2011. *Business Process Modeling Notation (BPMN), version 2.0.*

Smith, R. M., and S. T. Kishor. 1988. "Performability Analysis: Measures, an algorithm, and a Case Study". *IEEE Transactions on Computers* 37 (4): 406–417.

Tumay, K. 1996. "Business process simulation". In *Proceedings of the 1996 Winter Simulation Conference*, edited by J. M. Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain, 93–98. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

van der Aalst, W., J. Nakatumba, A. Rozinat, and N. Russell. 2010. "Business Process Simulation: How to get it right?". In *Handbook on Business Process Management*, edited by J. vom Brocke and M. Rosemann, 317–342: Springer-Verlag.

van der Aalst, W. M. P. 2010. "Business Process Simulation Revisited". In *Enterprise and Organizational Modeling and Simulation*, edited by J. Barjis, Volume 63 of *Lecture Notes in Business Information Processing*, 1–14. Springer Berlin Heidelberg.

van der Aalst, W. M. P. 2013. "Business Process Management: A Comprehensive Survey". *ISRN Software Engineering* 2013:1–37.

Vanderfeesten, I., J. Cardoso, J. Mendling, H. Reijers, and W. M. P. van der Aalst. 2007. "Quality Metrics for Business Process Models". In *BPM and Workflow Handbook*, edited by L. Fischer, 179–190. Future Strategies Inc.

AUTHOR BIOGRAPHIES

PAOLO BOCCIARELLI is a postdoc researcher at the Enterprise Engineering Department of the University of Rome Tor Vergata (Italy). His research interests include software and systems engineering and business process management, specifically with regards to the areas of modeling and simulation and model-driven development. His email address is paolo.bocciarelli@uniroma2.it.

ANDREA D'AMBROGIO is associate professor at the Enterprise Engineering Department of the University of Roma TorVergata (Italy). His research interests are in the fields of model-driven software engineering, dependability engineering, distributed and web-based simulation. His email address is dambro@uniroma2.it.

ANDREA GIGLIO is a Ph.D student at the Enterprise Engineering Department of the University of Rome "Tor Vergata" (Italy). His research interests include model-driven system and software engineering and business process management. His email address is andrea.giglio@uniroma2.it.

EMILIANO PAGLIA is a Ph.D student at the Enterprise Engineering Department of the University of Rome "Tor Vergata" (Italy). His research interests include model-driven engineering and business process modeling and analysis. His email address is emiliano.paglia@uniroma2.it.