A MSaaS PLATFORM FOR BUSINESS PROCESS MODELING & SIMULATION

Paolo Bocciarelli
Andrea D'Ambrogio
Matteo Maria Emanuele Cialei

Dept. of Enterprise Engineering
University of Rome Tor Vergata
Rome, ITALY
{paolo.bocciarelli,dambro,mcialei}@uniroma2.it

ABSTRACT

MSaaS (Modeling & Simulation as a Service) is an emerging paradigm in which service orientation and cloud computing infrastructures are jointly exploited to carry out M&S efforts. A MSaaS platform defines an ecosystem of resources that gives users the opportunity to manage the lifecycle of M&S studies. While MSaaS approaches can be effectively applied in various application domains, this paper focuses on the use of MSaaS for enabling the simulation-based analysis of business processes. In this context, this paper introduces BPSaaS (Business Process Simulation as a Service), a MSaaS platform for managing the various steps of a M&S study applied to the analysis of BPs: process modeling, model parameterization, model implementation, model execution and output analysis. To illustrate the features offered by the proposed platform, the paper includes an example application that specifically focuses on the parameterization step and shows how a given BPMN model, edited with a BPMN-compliant tool, can be retrieved and annotated by use of the BPSim standard, and then conveniently transformed into a format ready to be executed on a business process simulation engine.

Keywords: BPMN, MSaaS, BPSim, business process modeling, business process simulation, model transformation.

1 INTRODUCTION

MSaaS (Modeling & Simulation as a Service) is recently emerging as an effective paradigm to bring the advantages of service-oriented architectures and cloud computing in the Modeling & Simulation (M&S) field (Siegfried et al. 2018).

The use of MSaaS platforms goes far beyond what is commonly denoted as *cloud simulation* (Fujimoto 2016), which refers to the use of cloud accessible resources to execute a simulation model in either a centralized or distributed fashion.

A MSaaS platform defines an ecosystem of resources that gives users the opportunity to properly manage the whole lifecycle of a M&S effort. The main objective of MSaaS platforms is to improve accessibility and interoperability of data and resources commonly used in a M&S study (e.g., simulation models, experimental frames, simulation engines), with a significant impact in terms of improved effectiveness of M&S applications, which are shown to get significant savings in terms of cost, time, manual effort and required know-how (Siegfried et al. 2018).

In particular, this paper addresses the use of MSaaS platforms for the simulation-based analysis of business processes (BPs).

BP analysis is an important activity carried out in different phases of the BP lifecycle, e.g., at BP definition time, to predict the behavior of the *to-be* process, or at BP execution time, when the *as-is* process experiences performance downgrades to fix.

A BP model is typically used to represent a BP at a given level of abstraction, so to capture the relevant details for the specific scope. Then, the model is used to verify the properties of the BP under analysis. Due to the complexity of real-world BPs, using analytical approaches that provide efficient closed form solutions is often unfeasible, and thus the BP analysis is conveniently carried out by simulating the BP model, that is by producing a software implementation of the BP model that can be run to execute the model over time and get insights about the BP behavior.

The process modeling language currently adopted as a de-facto standard is *BPMN (Business Process Model & Notation)*, which is released by the Object Management Group (OMG 2014). BPMN popularity is mostly due to its ease-of-use and versatility, which make the language understandable and profitable to users with different know-how, from business analysts down to IT developers in charge of implementing BPs specified in BPMN.

In order to be analyzed, a BPMN model needs to be properly annotated with data that define the so-called experimental frame, in other words the conditions under which the process is observed or experimented with (Zeigler et al. 2018). These data contribute to collect the *simulation parameters* that allow using the annotated BPMN model as a simulation model that is to be translated into a computer-executable format, which is finally run to get the results of interest.

In this respect, this paper introduces *BPSaaS (Business Process Simulation as a Service)*, a MSaaS platform for managing the various steps (i.e., process modeling, model parameterization, model implementation, model execution, output analysis) of a M&S study applied to the analysis of BPs .

A distinctive feature of the BPSaaS platform is the provision of services that contribute to satisfy the FAIR (Findability, Accessibility, Interoperability and Reusability) properties of MSaaS platforms, with a specific focus on the interoperability of heterogeneous data and resources used in a M&S study for BP analysis. Such services exploit principles and standards introduced in the model-driven engineering field.

To give an example of the features offered by the BPSaaS platform, the paper specifically focuses on the parameterization step and shows how a given BPMN model, edited with a BPMN-compliant tool, can be retrieved and annotated by use of the BPSim standard (Workflow Management Coalition: 2016), a BPMN extension to specify simulation-related information. The model annotated with BPSim is then conveniently reused to generate a model that utilizes a different BPMN extension for the specification of simulation parameters. The paper addresses PyBPMN (Bocciarelli and D'Ambrogio 2011), a BPMN extension that allows the automated execution of parameterized models on the eBPMN simulation engine (Bocciarelli et al. 2014). A case study is used as a running example to illustrate how the BPSaaS platform can be applied to carry out the aforementioned workflow.

The paper is structured as follows: Section 2 provides background information that allows the reader to get a better understanding of the paper contribution. Section 3 describes the architecture of the BPSaaS platform, while Section 4 illustrates an example application to the parameterization of a BPMN-based simulation model. Section 5 discusses the paper contribution and, finally, Section 6 provides concluding remarks.

2 BACKGROUND

This section describes the main technologies and standards that are used in the proposed approach, namely *PyBPMN*, the BPMN extension that is used to annotate BPMN models so to automatically generate the corresponding simulation code, *eBPMN*, the framework used to specify and execute the simulation code, and *BPSim*, the standard used to define the parameters for BPMN model simulation.

2.1 PyBPMN and eBPMN

The Business Process Modeling and Notation (BPMN) (OMG 2014) is a visual language promoted by the Object Management Group (OMG) for the specification of business processes. Among the other advantages deriving by its adoption, BPMN allows the specification of graphical process models that are easily readable by all the involved stakeholders. As its popularity has rapidly grown over the years, BPMN has became a de-facto standard in the business process management area.

Despite its pervasiveness and completeness, BPMN has limitations in the specification of non-functional properties of a BP, as underlined in (Saeedi et al. 2010). To address such issues, several authors have provided extensions of the standard BPMN language.

In this context, in previous contributions we proposed a BPMN extension, namely PyBPMN (Performability-enabled BPMN), which addresses the specification of performance and reliability properties of a BP (Bocciarelli and D'Ambrogio 2011, Bocciarelli and D'Ambrogio 2014, Bocciarelli et al. 2016).

The PyBPMN extension addresses four main areas of non-functional properties, to specify the workload associated to the whole business process or to single tasks, the performance and reliability properties associated to the process or to single tasks, and the actual resources used to execute process tasks. PyBPMN allows users to define non functional properties for either atomic resources or groups of resources consisting of concurrent or alternative resources. A PyBPMN resource can model a human worker, an equipment, a functional division of an organization, an autonomous system, a web service, or any other entity which can be used to execute process tasks.

As further clarified in Section 3.4, PyBPMN is used in this work to annotate BPMN models so to drive the automated generation of the executable BP simulation code in eBPMN, a Java-based domain-specific simulation framework that has been defined to comply with the execution semantics of the BPMN 2.0 specification. Under an implementation point of view, eBPMN is integrated on top of the SimArch layered discrete-event simulation (DES) architecture (Gianni et al. 2011).

eBPMN implements a token abstraction to simulate the execution of a BPMN process instance. A token is generated by a `Start` node to traverse the process nodes according to the specific sequence flows defined by the BPMN model. Each eBPMN element handles the token according to its execution semantics. At the `End` node the token is destroyed and the simulation environment gathers information about the process instance execution. The time interval between two subsequent token generation events follows the probability distribution specified by the given workload characterization.

The mapping of a BPMN model annotated by use of the PyBPMN extension to the corresponding eBPMN implementation is supported by introducing appropriate automated model transformations (Bocciarelli et al. 2014).

In this paper, eBPMN is used for specifying the executable simulation code of a BP under study.

2.2 BPSim

The Business Process Simulation (BPSim) (Workflow Management Coalition: 2016) is a standard framework introduced by the Workflow Management Coalition (WfMX) which allows the extension of process models specified in BPMN or XPDL with information required for conducing a simulation-based analysis.

The BPSim specification includes an UML-based metamodel, which defines the entities extending the BPMN/XPDL elements to capture the simulation-related information, and an interchange format defined by use of an XML Schema Definition (XSD).

BPSim organizes the analysis data in different *scenarios*, where each scenario is specified to address a given objective. As an example, the captured data can be *parameters* to be used for carrying-out a simulation-based analysis, *results* obtained in a previously conducted analysis, or *historical data* gathered from past real world executions of the process.

Specifically, BPSim parameter that refer to specific BPMN elements can be of *time*, *control*, *resource* or *cost* type. Moreover, *property parameters* allow the specification of additional properties assigned to BP elements, while *priority parameters* describe the priority of BP elements and whether the execution of BP elements is interruptible.

The reader is sent to (Workflow Management Coalition: 2016) for a complete description of the BPSim specification.

3 BPSAAS ARCHITECTURE

The following subsections give a detailed description of the BPSaaS platform, along with the principles that drive its implementation.

Specifically, Section 3.1 summarizes the BPSaaS requirements, Section 3.2 outlines the abstract architecture which identifies the capabilities BPSaaS shall provide, Section 3.3 describes the concrete implementation and the flow of activities through which the simulation-based analysis is carried-out, and, finally, Section 3.4 details how the BP specification can be translated to a given notation for simulation execution.

3.1 BPSaaS Principles and Requirements

In order to drive the design and development effort of BPSaaS, the following main requirements have been identified:

1. **MSaaS compliance**: according to the addressed architectural paradigm, the BPSaaS architecture shall exploit service orientation. The relevant services orchestration shall be founded on concrete services available in a cloud-based infrastructure;
2. **modeling capability**: BPSaaS shall provide appropriate features for retrieving or building BPMN models of the processes under study;
3. **parameterization capability**: according to its core objective, BPSaaS shall provide an automated support for parameterizing a BPMN model according to the BPSim standard, in order to effectively support the process simulation;
4. **mapping capability**: BPSaaS shall provide automated procedures to map the BPSim model to the notation suitable adopted by the specific simulation engine used to execute the BP simulation;
5. **simulation support**: BPSaaS shall provide concrete services for gathering and analyzing simulation output.

In the past we have investigated possible architectural approaches which exploit the MSaaS paradigm to ease the execution of M&S exercises. The result of such effort has been the identification of a MSaaS Reference Architecture (MRA), which aims at providing guidance for the implementation of MSaaS-based supporting tools (Bocciarelli et al.).

The BPSaaS abstract architecture, which is discussed in the next Section, is based on the aforementioned MRA.

3.2 BPSaaS Abstract Architecture

The BPSaaS abstract architecture, which is provided in Figure 1, includes three different layers. The first one is the *User Layer*, which is introduced for describing the platform from the users perspective: at this layer, a concrete architecture should include the specification of use cases with roles and responsibilities of each user. For the sake of conciseness, this layer is not further discussed in this work. The second layer is the *MSaaS Service Composition Layer* for i) the identification of candidate *abstract* services that provide the required capabilities and ii) the specification of the related *orchestration* upon which the subsequent implementation will be based. Finally, the *Service Implementation Layer* deals with the actual development, deployment, provisioning and integration of concrete services available in the underlying cloud-based infrastructure, according to the identified abstract orchestration. In this respect, it should be underlined that the Service Composition Layer identifies a set of abstract capabilities which a concrete system implementation shall provide.

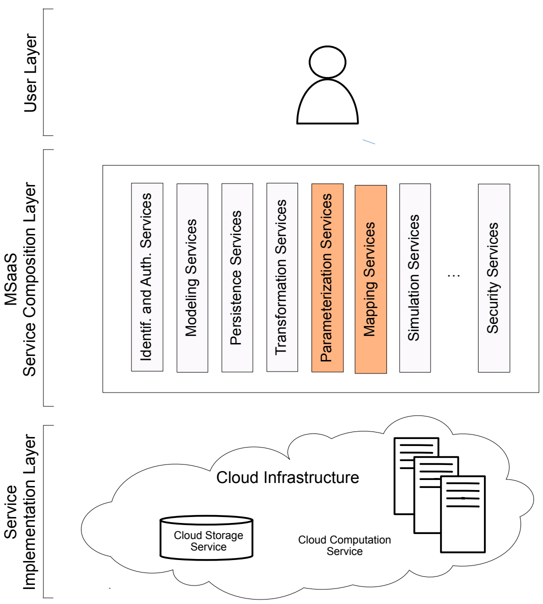

Figure 1: MSaaS abstract architecture.

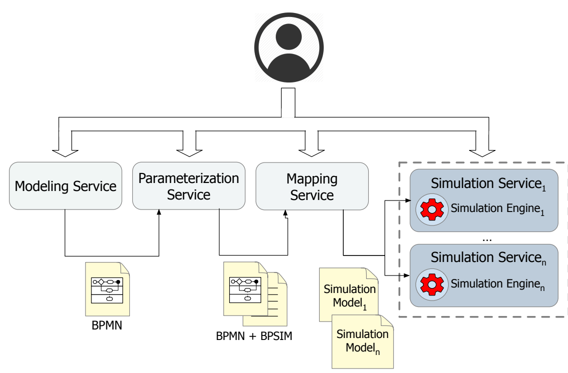

Figure 2: Simulation-based analysis of BPs by use of the BPSaaS platform.

In this work we specifically address the parameterization of BP simulation models, as well as their simulation on different concrete simulation services. As such, the BPSaaS abstract architecture extends the aforementioned MRA by explicitly considering the following additional services introduced for ensuring the compliance with requirements outlined in Section 3.1:

- a *Parameterization Service*, which allows the annotation of a BPMN model with concrete simulation parameters, such as: workload characterization, specification of the concrete resources responsible of executing process activities, service demand for each activity, and so forth;
- a *Mapping Service*, which allows the translation of the parameterized process model to an executable format, according to a specific simulation engine.

3.3 BPSaaS Concrete Implementation

The previous sections has outlined the abstract principles which have guided the BPSaaS development. This section completes the BPSaaS description by providing details about i) the flow of activities trough which the M&S-based analysis of a BP is carried-out and ii) the concrete technologies which have been used for BPSaas implementation.

Specifically, Figure 2 depicts the flow of activities trough which the M&S-based analysis of a BP is carried-out, as better specified below:

1. The first step deals with the specification of a process model. In this respect, the adopted notation is BPMN. The BP model can be either retrieved/edited by use of specific BPSaaS services or specified offline by using any BPMN-compliant modeling tool with XML export features. In either case, an XML-based serialization of the BPMN model is available.
2. At the second step, the BPMN model has to be parameterized in order to specify: i) the characteristics of the resources in charge of executing the process activities and ii) the non-functional properties of the BP. In this respect, BPSaaS provides a parameterization service which implements the relevant abstract capabilities shown in Figure 1. The output of this step is constituted by a BPSim description associated to the BPMN model. Both such artifacts are serialized as XML files.
3. At the third step, the *mapping service* is executed. According to its requirements, BPSaaS is not tied to a specific simulation execution notation or tool. Differently, in order to provide the highest flexibility degree, a *translation* step is carried-out that maps the BPSim model to an appropriate formalism, suitable for being executed by the concrete engine provided by the relevant simulation service. As further clarified in Section 3.4, the concrete mapping service implementation hereby adopted generates the eBPMN-based Java code which specifies the BP simulation code.
4. At the last step, the simulation model is given as input to the service which implements the concrete simulation engine. Then, the BP is finally simulated and the results are collected to be then analyzed by the user.

3.4 Mapping of BPSim to eBPMN

The flexibility of the BPSaaS platform allows users not to be locked-in to the use of a specific BP simulation tool. Differently, any simulation engine could be virtually used, provided there is an existing *mapping service* able to translate the BPSim specification to the relevant simulation notation adopted by the given simulation engine.

As stated in Section 1, in order to show a concrete application of the BPSaaS platform, this work assumes the availability of an eBPMN-based simulation engine.

Due to its proven effectiveness to support the software development effort (Zacharewicz et al. 2020, Bocciarelli and D'Ambrogio 2014, D'Ambrogio et al. 2016), the implementation of the mapping service is inspired by a model-driven engineering approach.

Table 1: Mapping of data types.

BPSim	PyBPMN
BooleanParameter	NFP_Boolean
DateTimeParameter	NFP_DateTime
DurationParameter	NFP_Duration
FloatingParameter (Float)	NFP_Real
FloatingParameter (TimeUnit)	NFP_DateTime
NumericParameter	NFP_Integer
StringParameter	NFP_String
DistributionParameters	GaWorkloadEvent

Table 2: Mapping of performance-related parameters.

BPSim	PyBPMN
TimeParameters.setup TimeParameters.processing TimeParameters.validation TimeParameters.rework	PaService.serviceTime (aggregated value)
TimeParameters.duration	PaService.service

Specifically, principles and standards introduced by the OMG's Model-driven Architecture (OMG 2003) have been adopted to carry-out the mapping of BPSim to eBPMN throughout two steps:

- **Generation of a PyBPMN model**: eBPMN has been designed as a framework for the straightforward simulation of BPs specified by use of PyBPMN. Thus, a preliminary *model-to-model* transformation is executed to map the BPSim specification to the corresponding PyBPMN model. The transformation is specified in the QVT Operational Mappings language introduced by the QVT (Query/View/Transformation) standard (OMG 2016).
- **Generation of the eBPMN code:** a *model-to-text* transformation is then carried-out to generate the corresponding eBPMN code. The transformation, which is fully described in (Bocciarelli et al. 2014), is specified in the MOF Model-to-Text (MOFM2T) language (OMG 2008), while the related implementation is based on Acceleo (Eclipse Foundation 2012), available as a plugin of the Eclipse platform.

As regards the generation of the PyBPMN specification from a BPSim model, it should be noted that BPSim and PyBPMN describe the non-functional properties of a BP from different perspectives. BPSim addresses the specification of non-functional properties from a broader perspective. It addresses the several parameters which might affect the process simulation. As an example, BPSim allows the configuration of: i) a *simulation scenario* (e.g., giving the opportunity to define the number of simulation iterations or the seed used for pseudo-random number generation), ii) the *time-related parameters* associated to each activity (e.g., the service time, the waiting time, etc.), iii) the *resources* characterization (e.g., the resource availability according to a given calendar). Differently, PyBPMN focuses on the detailed specification of the performance- and reliability-related parameters which affect the process execution (e.g., the resources MTTF and MTTR parameters, or the service demand according to a given probability distribution). In this respect, the mapping service adopted in this work takes into consideration only those BPSim elements that have a corresponding element in PyBPMN.

In order to outline a few mapping details, Table 1 shows how BPSim data types are translated to corresponding PyBPMN elements, while Table 2 describes the mapping of performance-related properties.

4 EXAMPLE APPLICATION

This section discusses an example application of the BPSaaS platform to the M&S-based analysis of a BP. The example hereby discussed refers to an emergency attendance process that is enacted when a request for assistance is received by an emergency dispatcher.

The choice of the emergency treatment process is motivated by the fact that it has been already investigated in previous works (Bocciarelli et al.), which focus on an in-depth analysis carried out by use of an eBPMN-based model-driven approach. Differently, in this paper, the same process has been used to show

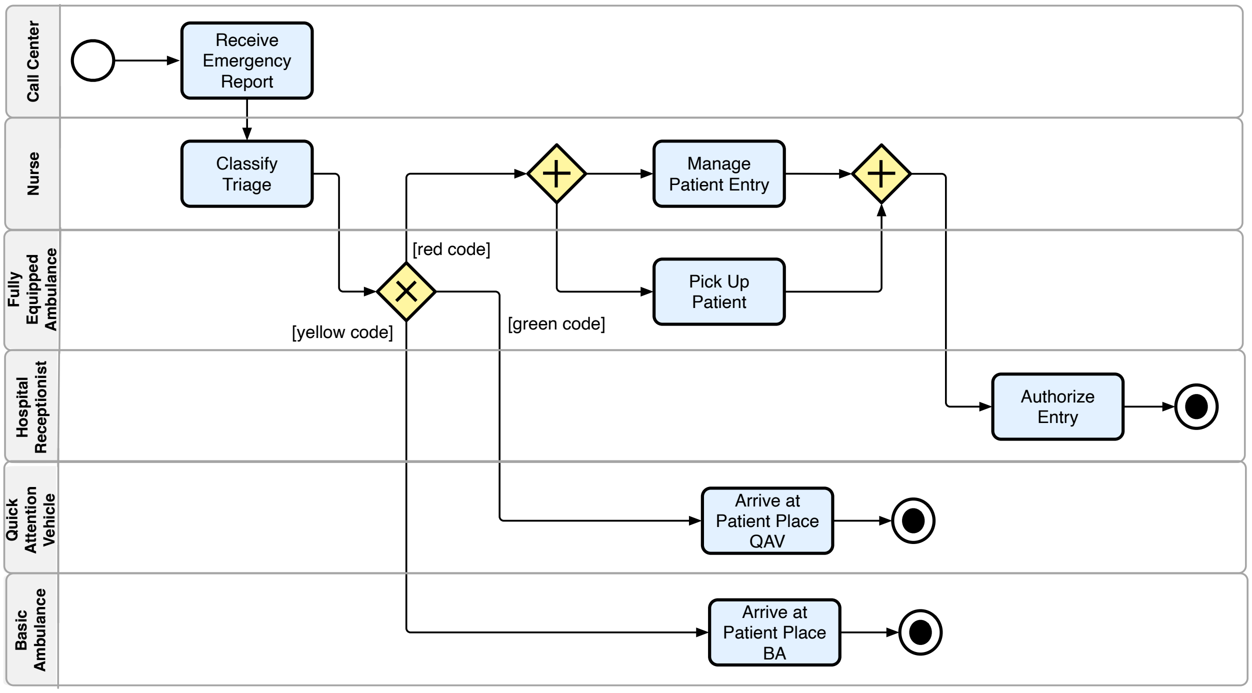

Figure 3: BPMN model of the emergency attendance process.

how BPSaaS can be effectively used for enacting the M&S-based analysis of a BP according to a MSaaS perspective, and using the services provided by the proposed platform: a modeling service, a parameterization service, a mapping service and, finally, a simulation execution service, as described in the following subsections.

4.1 BP Model Specification

The emergency attendance process flow is structured as follows: when a new emergency report is received, the *call center agent* is responsible of gathering all the information about the emergency. A report is then transmitted to a *nurse* who determines the relevant *severity* degree by assigning one of the following codes:

- **green code**: low severity, the patient can treated on spot and no urgent intervention is required;
- **yellow code**: medium severity, the patient requires some special attention. The emergency might require to be promptly faced but the treatment can be provided on spot;
- **red code**: high severity, the patient must be taken to an hospital.

The severity degree determines the subsequent treatment process: a green code emergency is assisted by a quick attention vehicle (QAV). Yellow codes emergencies require the intervention of a basic ambulance (BA). Finally, in a red code emergency, the patient needs to be taken to the nearest hospital for receiving appropriate treatments, so a fully equipped ambulance is needed. Moreover, during the transfer, a nurse prepares the paperwork needed for admitting the patient to the hospital's emergency department.

The BPMN model representing the process under study is depicted in Figure 3. The probabilities of alternative flows have been determined by analyzing historical data of past emergency reports.

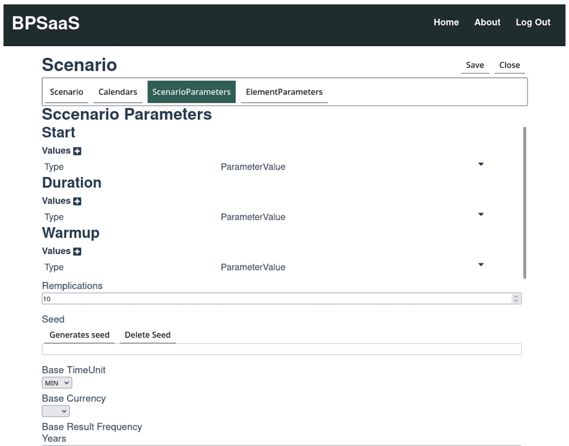

Figure 4: Specification of the *Scenario* parameters.

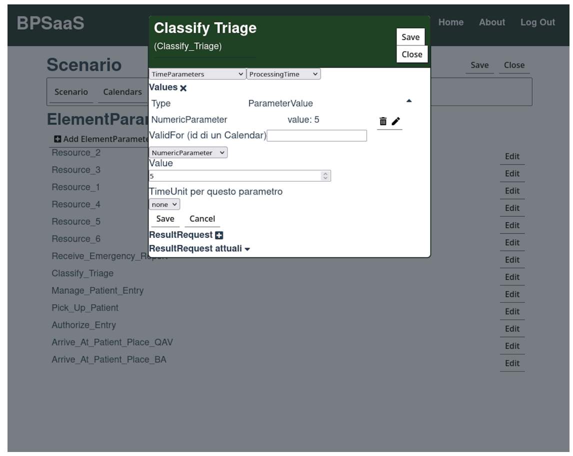

Figure 5: Specification of the service time for the *Classify Triage* activity.

As mentioned in Section 3.3, the BPMN model can be edited using either a simulation service (see Figure 2) or any BP modeling tool providing XML-based serialization features. The example application thus assumes the availability of an XML-based serialization of the emergency attendance process model, regardless the specific concrete service/application used for its editing.

4.2 Model Parameterization

The model parameterization is carried-out by using the relevant service provided by the BPSaaS platform. The service, as shown in Figure 4 and Figure 5, allows analysts to first upload the BPMN model of the BP under study (serialized as an XML file) and then specify the required simulation parameters according to the BPSim standard. The output of the parameterization step is a BPSim model describing the properties required for the process simulation.

It's worth noting that, as described in Section 3.4 and further clarified in Section 4.3, this paper assumes the adoption of an eBPMN simulation engine. Thus, the BPSim model is first translated into a PyBPMN specification, which is eventually used for generating the Java based eBPMN code implementing the executable simulation. In this respect, the model parameterization focuses on those BPSim parameters that have a corresponding representation as PyBPMN elements (mostly time-related parameters).

4.3 Model Transformation

According to the BPSaaS rationale, the model transformation step deals with the generation of an executable process representation to be given as input to the adopted simulation engine. In this paper case, the model transformation step is carried out in two sub-steps:

1. a model-to-model transformation is executed for translating the BPSim model into a PyBPMN specification;
2. the so-obtained PyBPMN specification is given as input to a model-to-text transformation for generating the relevant eBPMN Java code implementing the executable simulation.

4.4 Simulation Execution

The last step of the BP analysis consists in the actual execution of the simulation model implementation obtained at the previous step. The simulation results allow analysts to determine whether or not the process behavior meets the users' expectation and, in the negative case, identify possible improvements. In this respect, the BPSaaS platform provides a service that gives access to the eBPMN simulation engine used to execute eBPMN code.

In order to show an example of performance indices that the simulation-based analysis contributes to obtain, Table 3 shows the average process execution time (i.e., the average time required to complete a process instance execution) for different values of the emergency request inter-arrival rate. An in-depth analysis of the Emergency Attendance process is out of the scope of this paper, which instead aims at showing how the BPSaaS platform effectively enables M&S-based analysis of BPs according to a MSaaS perspective. For a more detailed discussion of the Emergency Attendance process simulation, interested readers are referred to (Bocciarelli et al.).

Table 3: Processing Time for the Emergency Attendance Process.

Emergency Request Interarrival Rate (min^{-1})	**Red Code Process Time (min)**
0.025	36.70
0.05	36.94
0.1	37.69
0.15	39.53
0.2	44.51
0.25	62.30
0.275	122.67
0.3	1012.82

5 DISCUSSION

As mentioned in Section 1, a distinctive feature of the BPSaaS platform, inherited by MSaaS approaches, is to provide services that contribute to improve the FAIR properties in simulation-based BP analysis efforts, with a specific focus on the interoperability issue, thanks to the provision of services exploiting principles and standards introduced in the model-driven engineering field.

The lack of interoperability is particularly affecting the business process modeling and simulation field, in which tool lock-in and the limitations of single-user tools are widely recognized as issues that do not enable business analysts to fully exploit the potential of M&S techniques (Gao and Chen 2014).

Various tools are currently available to edit and possibly simulate a BPMN model of a BP, with some tools (Bizagi, Signavio, SmartProcess, just to mention a few) also offering access through external APIs, which contribute to replace manual coding activities with simple and familiar web interfaces. However, what they offer cannot be easily reused or operated in a different tool or environment. This is even exacerbated when it comes to the several BPMN extensions that have been proposed to improve BPMN expressiveness (Zarour et al. 2019).

The provision of model transformation services in the BPSaaS platform contributes to the definition of an open environment that allows users to overcome the aforementioned issues. Beyond the specific implementation technology that is used to implement the platform (large-grained services or applications vs. fine-grained microservices, containers vs. virtual machines, etc.), it is the model-driven architectural design that makes the BPSaaS platform easily extensible to accommodate additional needs or integrate different approaches.

This paper explicitly deals with parameterization approaches, yet similar results can be achieved to deal with, e.g., different simulation engines and/or BP modeling languages. The combination of model-driven, largely automated, approaches with service-orientation and cloud computing opens the way to significant improvements in terms of accessibility of M&S resources in the business process management field, in addition to the already proven advantages of MSaaS platforms (Wainer and Wang 2017, Gütlein and Djanatliev 2020, Zehe et al. 2015, Taylor et al. 2015).

6 CONCLUSIONS

This paper has introduced BPSaaS, a MSaaS platform for simulation-based BP analysis. The proposed platform has been designed to exploit model transformations based on principles and standards introduced in the model-driven engineering, so to provide a significant degree of automation and interoperability. The availability of model transformation services contributes to the definition of an open environment that can be easily extended for accommodating additional needs or integrating different approaches. A case study dealing with the annotation of simulation parameters on a BPMN model and the automated transformation into the corresponding model implementation has been used to illustrate the potential of the proposed platform. Work is in progress to extend the platform with additional transformation and support services, so to further improve the effectiveness and flexibility of the proposed approach.

REFERENCES

Bocciarelli, P., and A. D'Ambrogio. 2011. "A BPMN Extension for Modeling Non Functional Properties of Business Processes". In *Proceedings of the Symposium on Theory of Modeling and Simulation*, DEVS-TMS '11, pp. 160–168. Boston, Massachusetts.

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

Bocciarelli, P., A. D'Ambrogio, A. Giglio, and E. Paglia. 2016. "A BPMN extension to enable the explicit modeling of task resources". In *CEUR Workshop Proceedings - 2nd INCOSE Italia Conference on Systems Engineering, CIISE 2016*, Volume 1728, pp. 40–47, CEUR-WS.

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*, MODELSWARD'14. Lisbon, Portugal, SciTePress.

Bocciarelli, P., A. D'Ambrogio, and U. Durak. "ArTIC-M&S: An Architecture for TOSCA-Based Inter-Cloud Modeling and Simulation". In *Proceedings of the 2020 Winter Simulation Conference (WSC)*.

D'Ambrogio, A., E. Paglia, P. Bocciarelli, and A. Giglio. 2016. "Towards performance-oriented perfective evolution of BPMN models". In *Proceedings of the 2016 Spring Simulation Multiconference*, The Society for Modeling and Simulation International (SCS).

Eclipse Foundation 2012. "Acceleo". https://www.eclipse.org/acceleo/, Accessed 29th Jan 2022.

Fujimoto, R. M. 2016, may. "Research Challenges in Parallel and Distributed Simulation". *ACM Trans. Model. Comput. Simul.* vol. 26 (4).

Gao, J. T., and W. Chen. 2014, 2. "Research on Collaborative Mechanism in Large and Complex Business Process Modeling". In *Advances in Applied Sciences and Manufacturing*, Volume 850 of *Advanced Materials Research*, pp. 1012–1015, Trans Tech Publications Ltd.

Gianni, D., A. D'Ambrogio, and G. Iazeolla. 2011. "A software architecture to ease the development of distributed simulation systems". *Simulation* vol. 87 (9), pp. 819–836.

Gütlein, M., and A. Djanatliev. 2020. "Modeling and Simulation as a Service using Apache Kafka". In *Proceedings of the 10th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, pp. 171–180.

OMG 2003. *MDA Guide revision 2.0 (ormsc/14-06-01)*. https://www.omg.org/cgi-bin/doc?ormsc/14-06-01.pdf, Accessed 29th Jan 2022.

OMG 2008. *MOF Model to Text Transformation Language (MOFM2T), 1.0*. https://www.omg.org/spec/MOFM2T, Accessed 29th Jan 2022.

OMG 2014. *Business Process Model And Notation (BPMN) version 2.0.2*. https://www.omg.org/spec/BPMN/, Accessed 4 Mar 2022.

OMG 2016. *Meta Object Facility (MOF) Query/View/Transformation, version 1.3*. https://www.omg.org/spec/QVT/1.3/, Accessed 29th Jan 2022.

Saeedi, K., L. Zhao, and P. R. F. Sampaio. 2010. "Extending BPMN for Supporting Customer-Facing Service Quality Requirements". *Web Services, IEEE International Conference on* vol. 0, pp. 616–623.

Siegfried, R., C. McGroarty, and T. van den Berg. 2018. "Modelling and Simulation as a Service: a new reality". *CSIAC Journal* vol. 6 (6), pp. 18.

Taylor, S. J. E., A. Khan, K. L. Morse, A. Tolk, L. Yilmaz, J. Zander, and P. J. Mosterman. 2015. "Grand challenges for modeling and simulation: simulation everywhere—from cyberinfrastructure to clouds to citizens". *SIMULATION* vol. 91 (7), pp. 648–665. https://doi.org/10.1177/0037549715590594.

Wainer, G., and S. Wang. 2017, 05. "MAMS: Mashup Architecture with Modeling and Simulation as a Service". *Journal of Computational Science* vol. 21, pp. 113–131.

Workflow Management Coalition: 2016. "Business Process Simulation Specification, version 2.0". https://www.bpsim.org/specifications/2.0/WFMC-BPSWG-2016-01.pdf, Accessed 19th January 2022.

Zacharewicz, G., N. Daclin, G. Doumeingts, and H. Haidar. 2020. "Model driven interoperability for system engineering". *Modelling* vol. 1 (2), pp. 94–121.

Zarour, K., D. Benmerzoug, N. Guermouche, and K. Drira. 2019, 11. "A systematic literature review on BPMN extensions". *Business Process Management Journal* vol. 26 (6), pp. 1473–1503.

Zehe, D., W. Cai, A. Knoll, and H. Aydt. 2015. "Tutorial on a modeling and simulation cloud service". In *Proceedings of the 2015 Winter Simulation Conference (WSC)*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, pp. 103–114. Piscataway, New Jersey, IEEE.

Zeigler, B. P., A. Muzy, and E. Kofman. 2018. *Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations, 3rd edition*. Academic press.

AUTHOR BIOGRAPHIES

PAOLO BOCCIARELLI is a postdoc researcher in the Department of Enterprise Engineering, University of Rome Tor Vergata, Italy. His email address is paolo.bocciarelli@uniroma2.it.

ANDREA D'AMBROGIO is associate professor of systems and software engineering in the Department of Enterprise Engineering, University of Rome Tor Vergata, Italy. His email address is dambro@uniroma2.it.

MATTEO MARIA EMANUELE CIALEI is a master student in the Department of Enterprise Engineering, University of Rome Tor Vergata, Italy. His email address is mcialei@uniroma2.it.