Towards BPM@*Runtime*

Luiz F. B. Loja[1], **Sofia L. Costa Paiva** [2], **Juliano Lopes Oliveira** [3]

[1]Instituto Federal de Goiás

[2]Departamento de Ciências da Computação, Universidade Federal de São João Del Rei

[3]Instituto de Informática, Universidade Federal de Goiás

luiz.loja@ifg.edu.br, sofia@ufsj.edu.br, juliano@inf.ufg.br

Abstract. *Producing software to manage Business Processes (BP) in Information Systems (IS) requires considerable effort and time. This paper describes a software component that uses a Model-Driven Engineering (MDE) approach to support BP management in IS into the context of an IS application framework. The BP Manager component allows the definition of BP models using a high-level language, that is integrated with others framework components to generate the IS applications. This component has important improvements compared to other BP management tools, such as support for collaborative modeling and support for the execution of empirical and ad hoc processes. Furthermore, the integration of the BP management component into the framework brings significant increases in productivity of IS software development and maintenance.*

1. Introduction

Software-intensive Information Systems (IS) collect, store, transform, and distribute information [Laudon and Laudon 2015]. IS are built to handle information linked to the organization's business, providing knowledge for supporting business decisions. IS software development includes, among other aspects, the specification and data modeling, functions, rules, and business processes (BP). Moreover, it also demands several ways to provide an easy interaction between users and the data system information.

A framework to support the development of IS domain-specific software based on high-level models has developed by the Software Engineering Research Group of Computing Institute of the Federal University of Goiás (INF/UFG). It supports the automatic generation and IS software maintenance using conceptual models, in which a considerable amount of IS software is generated using metamodels that describe aspects of IS related to business concepts, business rules, and user interfaces.

Despite the success achieved with this framework's application for agribusiness control, a limitation was identified that makes it difficult to generate a significant part of the IS software, which is related to the BP definition and execution. Two new metamodels was produced for the refactored framework [Loja et al. 2010, da Costa et al. 2010], and the BP component was implemented [Oliveira et al. 2011]. This paper describes the BP management system (BPMS) component developed for this framework, which may be used to manage, model, and execute BP in IS. Also, an experiment aims at to evaluate the tool's functionalities and its expression power are presented.

This paper is structured as follows. Section 2 presents the requirements for component creation. Section 3 describes the main component features. Section 4 relates an

experiment with potential users. Finally, Section 5 enhance final remarks and future work.

2. Requirements for BP Management Systems through Executable Models

BPMS tools allow one to model processes using, in general, the Business Process Modeling Notation (BPMN)[1]. A process is typically created and molded using a graphic editor. An execution machine instantiates and executes modeled processes, allocating tasks among process participants, and interacting with them to complete the process execution.

Several BPMS tools available on the market. Bizagi Bizagi[2], Bonita[3], ProcessMaker[4], and Activiti BPM[5] may be cited as instances of the state of practice [Sousa et al. 2018]. Bizagi and Bonita have regular BPMS features but they have some significant limitations. For instance, Bonita does not allow changing processes during its execution and does not support ad hoc and empirical process execution. Besides, Bonita has no features that allow the responsibilities distribution during the task execution and has several restrictions related to process sharing. Bizagi and Bonita do not support ad hoc processes execution, but it allows modeling it. They do not grant changing on the process flow during its execution, turning harder the execution of empirical processes. Also, Bizagi does not support responsibility distribution during the activities execution in a process. Both tools do not assist in modeling the collaborative process, once that responsible for process definition are unable to build processes using already defined process components. Therefore, it is not possible to employ process components already defined in other process definitions. Besides, several process standards are not implemented by these tools, such as ad hoc processes [Russell et al. 2006a].

Other works propose approaches using BP management based on BPMN [Awad et al. 2009, Tsai et al. 2007]. However, such works do not implement all process patterns, they do not support the reuse of processes components and the responsibilities definition. Finally, they could not solve known problems of BPMN notation [Van Nuffel et al. 2009]. The need of modern organisations to change and adapt BP in a increasing speed lead the community to address this problem with models@runtime, in which models reflects the system's current status at any point in time and allow adaptation mechanisms [Redlich et al. 2014]. However, models created using BPMN notation are not literally executable [Goldstein et al. 2019].

Based on the limitations identified in BPMS tools, the main requirements identified for the BP management component are described as follows.

1) Collaborative Modeling and Interaction. BPMS tools aim at supporting the processes modeling and execution, providing a collaborative environment [Loja et al. 2010]. Such an environment may provide a way for modeling processes in a collaborative form, i.e, smoothing the processes sharing and the processes components reuse previously defined. However, the process metamodels structure adopted in existing tools turns the sharing process component harder. For instance, the metamodel does not allow to share processes artifacts. In general, theses artifacts are defined in the context of

[1]http://www.bpmn.org/
[2]https://www.bizagi.com/solutions/intelligent-process-automation
[3]https://www.bonitasoft.com/
[4]https://www.processmaker.com/
[5]https://www.activiti.org/

a specific process. Thus, component sharing is done only at a process level, which limits the reuse of processes parts to define new processes.

2) Ability to Change Running Processes and Alteration. BP are flexible and changeable, being susceptible to quick and constant changes. A possibility to modify a process during its execution is an essential feature for BPMS tools, albeit this feature is not supported by the tools analyzed in this paper.

3) Definition and Execution of Distinct types of Processes. The ability to support distinct types of processes is important: empirical processes, which the task specification is made during the process execution; defined processes, which the definition tasks are entirely carried out before the process execution; and ad hoc processes. To deal with the intrinsic uncertainty in software development, Scrum, for instance, uses empirical processes to develop software products. The analyzed BPMS tools support only defined processes and are unsuitable for empirical processes. Ad hoc processes are formed by groups of activities that do not have sequence relations. It means that the activities execution flow order of an ad hoc process is undefined in BPMN[6]. The undefined order makes it harder to implement this type of process, due there is no predefined pattern to guide the activities execution within the process. To allow the modeling and execution of ad doc processes, it was used fundamental requirements to represent BP called Process Patterns [Russell et al. 2006b, van Der Aalst et al. 2003]. Expressing different sorts of process demands that the BPMS tool supports these patterns. Nevertheless, the analyzed tools do not implement all proposed patterns.

4) Distribution of tasks according to profiles. One of the difficulties that may happen during a BP execution is the responsibility allocation [de Mello and Rocha 2009]. In the current tools, only one participant may be allocated to perform each task, or, in the best scenario, a group of participants may be designated. Nevertheless, this group of participants would have the same role, all would be allocated as a task executor. The cited approach does not allow, for instance, to represent the corporation hierarchy chain involved in BP. During the execution of tasks defined in the BP, involved stakeholders may collaborate between themselves. A critical factor to achieve success in this cooperation is the selection and allocation of stakeholders with the appropriate profile to accomplish specific tasks. Participants' allocation must ensure that the necessary skills to perform the task are available on the designated team. Therefore, tasks must be distributed according to the stakeholders' profile (knowledge, experience, and skills) of each participant. It is critical that a participant that performs a task is not limited to its execution.

The COBIT model[7], for instance, allows the interaction between several roles, with distinct types of involvement, during process tasks execution. COBIT proposes four types of interaction profiles: responsible (executor), approver, interested, and consulted. Different interaction sorts define the distinct ways for the participant to get involved with a task, representing the reality of modern organizations more faithfully.

3. Business Process Management Overview

The software architecture of the BP manager component is composed of three main modules: the process metamodel; the process execution machine; and the interfaces that pro-

[6]http://www.bpmn.org/

[7]Control Objectives for Information and related Technology: http://www.itgi.or

vide the interaction with the user. The integration of these components to the IS development framework makes the IS composed process-oriented.

The BP metamodel [Loja et al. 2010] allows the reuse and sharing of Process Flow Objects, and it supports the modeling of ad hoc and empirical processes. The execution machine provides support for all other sorts of processes.

Figure 1 illustrates the initial screen of the component. In the upper corner of the screen, there are features related to the process modeling and execution, stakeholders and permissions. The available elements for navigation are on the left side. Figure 2 presents an enlargement of the toolbar in the BP component related to the modeling process, and a detailed view of the features associated with the process execution.

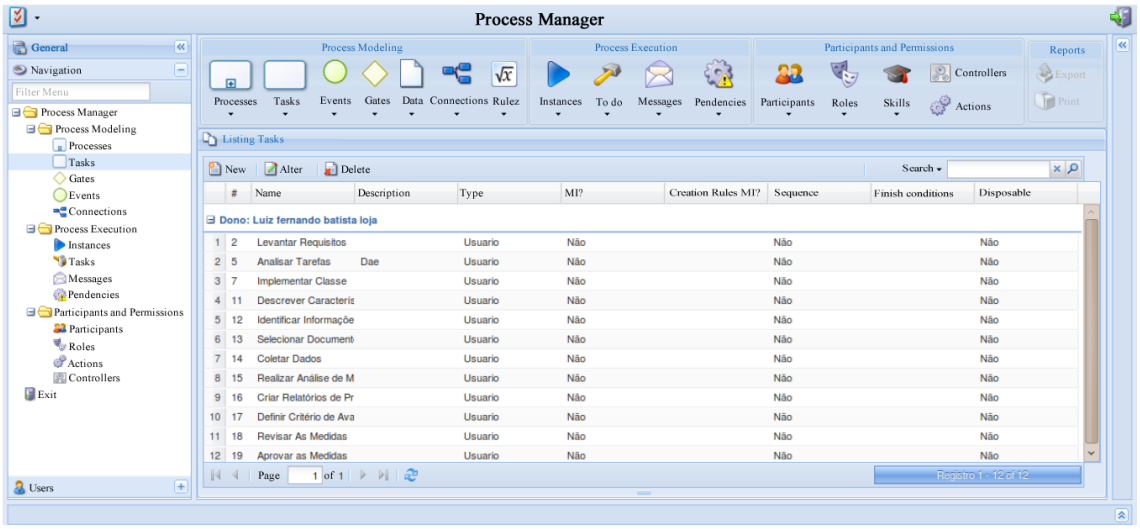


Figure 1. Main Screen.

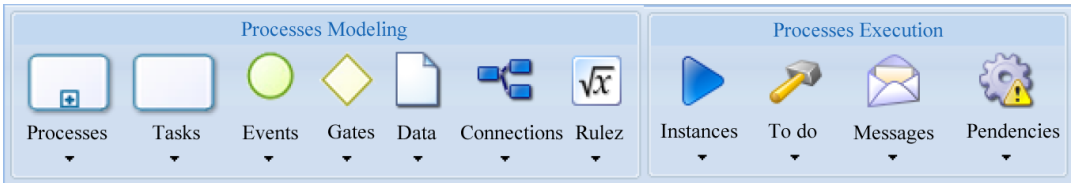


Figure 2. Top Menu of Process Modelling and Execution.

The component use involves a process with three main stages: creation, modeling, and BP execution. Features related to creation process include the ability to define processes, tasks, events, gateways, data objects, and connections. To provide a feature to reuse the parts of an already defined process in the definition of other processes, the first step involves the tasks elaboration that will be part of the BP, as well as the definition of its stakeholders, events, and gateways.

Instead of the BPMN notation (in which process elements are created in a specific process context), the reuse is possible due to component's process elements are context-free. Such a feature enables the reuse of process elements in the definition of other processes, and to share process elements among the participants of the process execution. Figure 3 illustrates the screen used to create a new process. There are similar interfaces to create events, participants, tasks, and gateways.

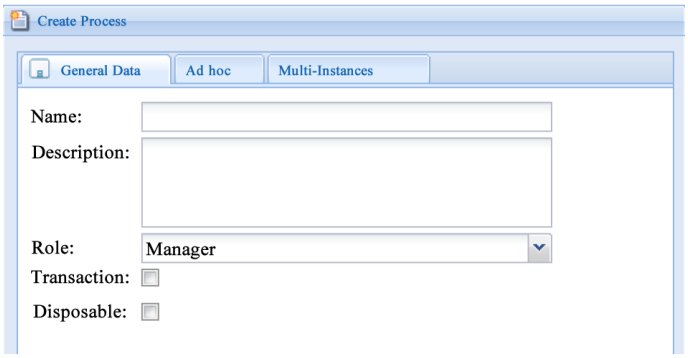


Figure 3. Process Creation Screen.

The component has a subset of BPMN notation that withdraws known limitation of BPMN notation [Van Nuffel et al. 2009]. Consequently, some BPMN artifacts are not available to modeling, such as the Text Annotation construction, which does not have an equivalent meaning in the real world, other than pools and lanes. In the metamodel [Loja et al. 2010], the role that a related stakeholder assumes in the task defines exactly who is responsible for it. From the created process and its constituent elements, begins the process modeling. All elements defined for the process, during the creation stage, as well as elements associated with previously defined processes are available for use in modeling. The component is equipped with a graphical modeling tool that allows the user to assemble a process connecting its elements graphically. Flow Objects compose a repository of process components, which can be reused to the design of new processes. Figure 4 shows a screenshot of the process graphical modeling tool with emphasis on the object repository available for the user.

Once that the process is modeled, it can be instantiated and executed. The process execution machine is responsible for instantiating and controlling the execution of any type of process defined according to the metamodel. An important feature of the machine is the possibility to change the process flow during its execution. This is possible thanks to the approach used in the machine based on tokens [Bendraou et al. 2009]. This approach allows the process to be executed without instantiating it in the application's memory.

To execute a process, the execution machine reads and performs a step at a time. This means that every time that a token is moved due to a process event, the machine analyses the token and define the next step in the process execution. Therefore, changes in the process may be instantly reflected in its flow, allowing the empirical processes modeling. Participants are defined at the time that the process instance is created. Tasks' responsibilities are allocated among these participants during the execution flow.

3.1. Potential Application of the BP Manager

A context that can benefit from the features provided by BP Manager Component is the software process improvement (SPI). Support for ad hoc and empirical processes is important, for instance, for organizations that are starting SPI projects, as they usually work in an ad hoc manner. On the other hand, the mature organizations, which seek to optimize their processes through experimentation, need to use empirical processes to evaluate costs and benefits of modifications applied to a process.

The graphic modeler and ad hoc process support are essentials to support the mat-

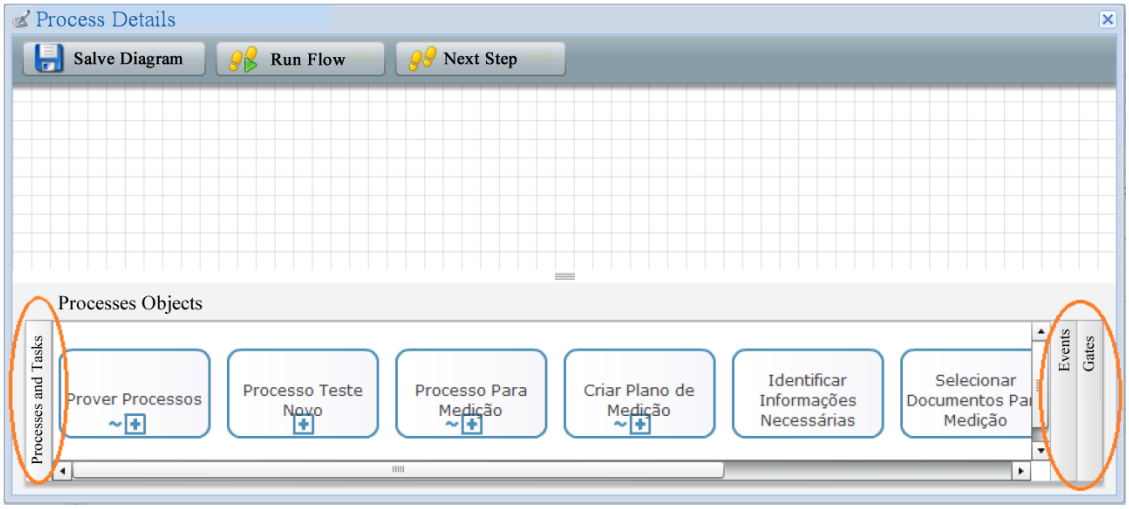


Figure 4. Graphical Modeler.

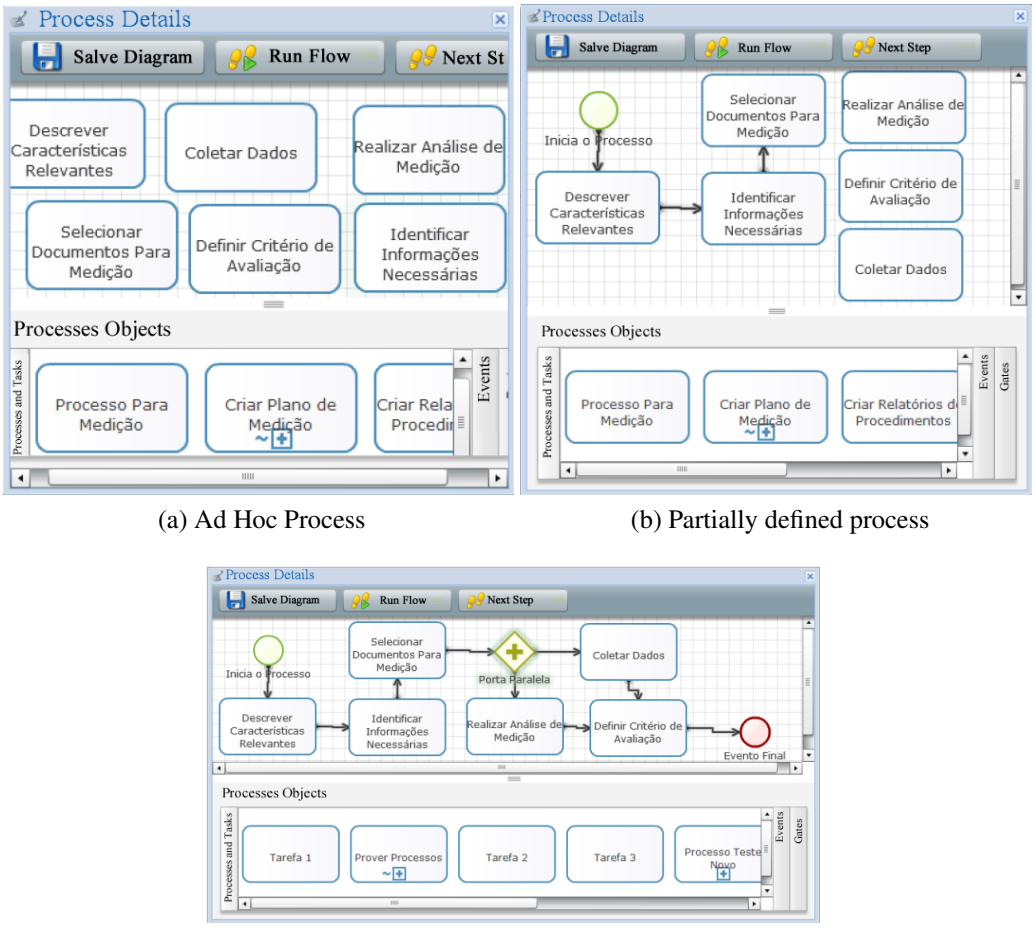
uration that an organization goes through during the first SPI initiatives. The sequence of scenes in Figures 6a, 6b, 6c illustrate three possible moments of the transition from ad hoc process to a defined process. 1) at the beginning tasks are known (they are available at the bottom side of the screen). However, your order is still arbitrary; 2) in an intermediary time the link among tasks is done, in such a way that it guides the process execution; and 3) finally the defined process specifies how it should be executed.

Another scenario of component application includes institutions that already defined their processes, nevertheless very few employees adopted them. In this case, the component may assist these employees, once it allows a remote interaction, using the internet, with employers. Also, it allows the employees to see the changes made on the process at the moment that he was not following up on the process execution. Such a feature allows all employees to have the most recent version of a given process, making it easy for collaborators to share organizational knowledge and allowing better monitoring and adhesion to the process execution.

The project team's resistance to adopt or modify a software process is related to the changing of organizational culture. Modifications in such culture, mainly in standard activities executed in an ad hoc manner, are hard to be accepted by those involved in SPI initiative. Therefore, in the SPI project execution, the older employees tend to be more resistant to the adoption of new processes.

The BP manager component may assist in the organizational changes by allowing the project manager to drive modifications in the process in a smooth and gradual manner. Support for ad hoc process execution allows the organization process evolution, starting with a simpler process and incrementing it over the time of the SPI project. Change stages in Figure 4 mitigate change impacts in the organizational culture, as they allow extended time of process maturation. Such extension allows employees to have more opportunities to absorb modifications, accept them, implement them, and institutionalize activities of a process, over the time where the process evolves and changes.

Processes evolve as a standard result of the organizational dynamic. The BP manager component may assist in this evolution. The ability that the component has to aid



(a) Ad Hoc Process (b) Partially defined process

(c) Defined process

Figure 5. Graphical Modeler.

in process modification during the execution of one of its instances allows evolving the process definition in a manner that it may adapt to the organizational reality evolution.

4. Experiment performed with the BP Manager Component

In order to evaluate the capacity of the BP Manager Component, it was conducted an experiment that aims at answer the following questions:

1. The component fulfill the proposed goals (requirements foreseen in section 2)?
2. Does it have enough expression power to accomplish the needs of a BPMS tool?

To evaluate the tool's functionalities and its expression power, it was defined procedures to collect data and how these data should be analysed. Both for the evaluation of the functionalities and for the evaluation of the power of expression of the tool, data collection procedures were defined and the way that obtained data should be analyzed. The experimental protocol initially forecast for the modeling and execution of a defined process. The process was described textually to the experiment participants and it was about a merchandise purchase involving customers, attendants and distinct payment forms. During the experiment, it was requested that the process be changed in different ways, for instance, to contemplate a change in the form of payment for the purchase.

Fourteen users who have basic knowledge about process and process modeling were selected. All of them were IT professionals, with higher education, and more than two years of experience in software development and/or people and projects management. Four participants had take part of a project to implement the ISO 9001 standard in the public institution service sector in which the experiment was applied. Selected users acted in different roles, including: systems analysts, developers and business analysts. To level the participants' knowledge in the experiment, a three-hour course on BP management and modeling was done, and on the use of the BP manager software component. The experiment was perform in three days. Each participant was able to finish the experiment in an average duration of one hour. Therefore, it was spend almost fourteen hours to perform all procedures, besides to the three hours of the level course. Then, the experiment was completely took in seventeen hours.

Table 1 shows that the tool obtained more than eighty percent of approval relative to the presence and well functioning of the evaluated features, evidencing that all features proposed were implemented. The categories that did not reach one hundred percent reflect that the participants did not understand the question or that they found a fail on the tool's operation that did not allow them to use a specific feature of the component. Regarding the expression power, each modeled process pattern were performed by the component's execution machine and the result was compared with the patterns behavior [van Der Aalst et al. 2003]. Nineteen of twenty patterns were implemented ensuring that the proposed metamodel is able to express a wide variety of flow process situations.

Feature	**Results**
Modifications in the process model during execution	96,42%
Modeling and execution of empirical processes	100%
Modeling and execution of ad hoc processes	85,71%
Responsibilities distribution to participants	85,71%
Sharing and reusing process elements	92,85%

Table 1. Responses by evaluated feature

5. Final Remarks

This paper presented a BP management software component able to offer a collaborative environment for BP modeling and execution in the context of a model-driven framework for IS generation. It supports the empirical and ad hoc process execution, and it implements the main process patterns identified in the literature. Also, it overcomes several limitations regarding the collaborative aspect in the BP modeling and execution. The component was developed to be accessed through the internet and was made in Ruby language with Rails framework besides the user interface was made using Flex.

The component's BP execution machine is able of managing the performance of any sort of process adhering to the defined BP metamodel. Moreover, the execution engine supports the interaction of more than one participant per activity. Such a feature allows until four types of different interactions based on the responsibility definition proposed in the COBIT model. An experiment to evaluate this component involving fourteen users which used the component to model and execute several BP. This is a part of a

framework in which has been maintained for more than six years by the Software Engineering Research Group of [Oliveira et al. 2011]. Such framework was refactored to add management features, BP definition and execution, both in terms of using the framework and in terms of code generation. This evolution allowed it to create process-oriented IS.

An improvement foreseen to the component involves the changing the BP model individually for each instance. Although the current component version allows the modification of BP models while they are being executed, if a model is changed, all its instances in use also will be modify. Therefore, this may generate some undesirable collateral effects. Moreover, the component does not impose restrictions on changing the BP models. A BP expression language adaptation is presented aims at providing support to the process models change restriction during the execution [Kowalkiewicz et al. 2008]. Thus, it would be possible to create a BP model that would be only partially modifiable. For instance, the definition of change or adaptation points in an institutionalized process. Such a feature would contribute to the optimization and extension of BP once the process could be used as a reference model for the BP instantiated from it. Another foreseen extension for the BP management component involves the version control of BP modification. This feature could store the change history of processes allowing the modeler to monitor the processes evolution defined and managed by the component.

Such a tool can also be adapted for coping with Systems-of-Information Systems (SoIS) requirements [Graciano Neto et al. 2017]. The BP execution engine already is able to change the process at runtime to reflect architectural changes a SoIS can be subject to. An extension on BPMN to represent multiple organizations rather than multiple departments of a single organization could finish the needed changes to offer (i) a BP execution engine to support design-time prediction of SoIS missions and a runtime monitoring of the SoIS flexible and interorganizational BP [Graciano Neto et al. 2019] and (ii) a BPMN extension to cope with this recurrent new architectural and organizational arrangements [Graciano Neto et al. 2018].

References

Awad, A., Grosskopf, A., Meyer, A., and Weske, M. (2009). Enabling resource assignment constraints in bpmn. *Hasso Plattner Institute, Potsdam*.

Bendraou, R., Jezéquél, J.-M., and Fleurey, F. (2009). Combining aspect and model-driven engineering approaches for software process modeling and execution. In *Intern. Conf. on Software Process*, pages 148–160. Springer.

da Costa, S. L., Graciano Neto, V., Loja, L. F. B., and de Oliveira, J. L. (2010). A metamodel for automatic generation of enterprise information systems. In *Anais do I CBSoft-I BW-MDD*, volume 8, pages 45–52.

de Mello, M. S. and Rocha, A. R. (2009). Gestão integrada da melhoria de processos em organizações de software. In *Anais do V Workshop Anual do MPS*, pages 34–41.

Goldstein, A., Johanndeiter, T., and Frank, U. (2019). Business process runtime models: Towards bridging the gap between design, enactment, and evaluation of business processes. *Inf. Syst. E-Bus. Manag.*, 17(1):27–64.

Graciano Neto, V. V., Horita, F. E. A., C., E., R., A. J., Hachem, J. E., Santos, D. S., and Nakagawa, E. Y. (2018). A study on goals specification for systems-of-information

systems: Design principles and a conceptual model. In *Proc. of SBSI, SBSI 2018, Caxias do Sul, Brazil, June 04-08, 2018*, pages 21:1–21:8.

Graciano Neto, V. V., Horita, F. E. A., Santos, R. P., Viana, D., Kassab, M., Manzano, W. A. E., and Nakagawa, E. Y. (2019). S.o.b. (save our budget): a simulation-based method for prediction of acquisition costs of constituents of a system-of-systems. *Revista Brasileira de Sistemas de Informação - iSys*, 12(4):6–35.

Graciano Neto, V. V., Oquendo, F., and Nakagawa, E. Y. (2017). Smart systems-of-information systems: Foundations and an assessment model for research development. In Araujo, R., Maciel, R., and Boscarioli, C., editors, *Grand Challenges in Information Systems for the next 10 years*, pages 1–12. SBC, Porto Alegre, Brazil.

Kowalkiewicz, M., Lu, R., Bäuerle, S., Krümpelmann, M., and Lippe, S. (2008). Weak dependencies in business process models. In *Intern. Conf. on Business Inform. Systems*, pages 177–188. Springer.

Laudon, K. C. and Laudon, J. P. (2015). *Management Information Systems: Managing the Digital Firm Plus MyMISLab with Pearson eText – Access Card Package*. Prentice Hall Press, Upper Saddle River, NJ, USA, 14th edition.

Loja, L. F. B., Neto, V. G., Costa, S., and Oliveira, J. (2010). A business process metamodel for enterprise information systems automatic generation. In *Anais do I CBSoft: Teoria e Prática-I BW-MDD*, volume 8, pages 37–44.

Oliveira, J. L., Loja, L. F. B., da Costa, S. L., and Graciano Neto, V. V. (2011). Um componente para gerência de processos de negócio em sistemas de informação. In *Proc. of VII SBSI*, pages 250–261, Salvador, Brazil. SBC. A component for business processes management in information systems (In Portuguese).

Redlich, D., Blair, G., Rashid, A., Molka, T., and Gilani, W. (2014). *Research Challenges for Business Process Models at Run-Time*, pages 208–236. Springer, Cham.

Russell, N., ter Hofstede, A., Van der Aalst, W., and Wohed, P. (2006a). On the suitability of uml 2.0 activity diagrams for business process modelling. In *Proc. of the 3rd Asia-Pacific Conf. on Conceptual Modelling 2006*, pages 95–104. Australian Comp. Soc.

Russell, N., Ter Hofstede, A. H., Van Der Aalst, W. M., and Mulyar, N. (2006b). Workflow control-flow patterns: A revised view. *BPM Center Report BPM-06-22, BPMcenter. org*, pages 06–22.

Sousa, M., Lopes, N., Ribeiro, O., and Silva, J. (2018). Evaluation of bpm tools open source/freeware. In *13th Iberian Conf. on Inform. Syst. and Techn. (CISTI)*, pages 1–6.

Tsai, C.-H., Luo, H.-J., and Wang, F.-J. (2007). Constructing a bpm environment with bpmn. In *11th IEEE Intern. Workshop on Future Trends of Distrib. Comp. Systems (FTDCS'07)*, pages 164–172. IEEE.

van Der Aalst, W. M., Ter Hofstede, A. H., Kiepuszewski, B., and Barros, A. P. (2003). Workflow patterns. *Distributed and parallel databases*, 14(1):5–51.

Van Nuffel, D., Mulder, H., and Van Kervel, S. (2009). Enhancing the formal foundations of bpmn by enterprise ontology. In *Adv. in Enterpr. Engg III*, pages 115–129. Springer.