

# BPMN 2.0 based modeling and customization of variants in business process families

Andrea Delgado, Daniel Calegari

Instituto de Computación, Facultad de Ingeniería

Universidad de la República

Montevideo, Uruguay, 11300

{adelgado, dcalegar}@fing.edu.uy

**Abstract**—Business processes may accept variants based on specific business requirements of an organization, leading to the definition of a process family. There are many proposals for the modeling of the common and variable parts of a process family, as well as to support the customization of each process variant (i.e., process configuration or tailoring). In this article, we present the results of a detailed study about the modeling and customization of process families based on the Business Process Model and Notation (BPMN 2.0) standard, and we introduce BPMNext, a novel approach devised for this purpose. The language is an adaptation of BPMN 2.0 based on the ideas provided by vSPEM, a language for the modeling of software processes families. We also present a supporting tool and its application to a real case study about the assignment of positions in exchange programs.

**Keywords:** Business process families, variability, Business Process Model and Notation (BPMN), vSPEM.

## I. INTRODUCTION

Business Process Management (BPM) [1], [2] offers a framework to support the definition, control and continuous improvement of business operation. In the context of medium and large scale, or distributed organizations, it is very common that business processes accept variants based on specific business requirements, e.g., different selling process depending on the kind of products or payment method, or different accountability steps depending on the country in which the process is executed.

In general, process modeling languages, e.g., the Business Process Model and Notation (BPMN 2.0 [3]), a standard language, do not explicitly support the specification of process families, i.e., a set of business processes which are based on a base process (also known as: customizable or base process) and a set of variation points which are elements of the base process than can be customized depending on the context.

However, there are many proposals [4], [5] which extend existent languages, or even provide a language independent way of modeling process families. The proposals basically follow one of these approaches: variability by restriction, or by extension. In the variability by restriction approach, there is a customizable process model containing every possible behavior (variants) and then such behavior is restricted (e.g., activities are skipped) to conform a final process variant. The customizable process model can be seen as the union or Least Common Multiple (LCM) of all process variants [4]. In the

variability by extension approach, the customizable process model only represents the most common, or shared, behavior between the process variants and then such behavior needs to be extended. In this approach, the customizable process model as the intersection or Greatest Common Denominator (GCD) of all process variants under consideration [4].

Some proposals come from the software process area in which the Software & Systems Process Engineering Meta-model Specification (SPEM 2.0 [6]) has a leading role. The language is a specific modeling language for software processes, involving concepts like disciplines, activities and tasks, work products, phases, delivery process, among others. Besides SPEM provides ways for variability modeling, it has some limitations. In this context, there is a variability extension for SPEM (v-SPEM, [7], [8]) which follows a very simple variability by restriction approach. The customization process in the software process area is called process tailoring.

In this article, we present the results of a detailed study about the modeling and configuration of process families based on BPMN 2.0. We also introduce BPMNext, an extension of BPMN 2.0 supporting the specification of process families. The approach is based on the ideas provided by the vSPEM approach for software processes. We also introduce a tool supporting the modeling of business process families based on BPMNext, which is an extension of the BPMN2 Modeler<sup>1</sup> within the Eclipse framework. We also apply this approach to an existent problem within our University with respect to the assignment of positions in academic exchange programs.

The rest of this paper is organized as follows. In Section II we provide the results of a detailed study about BPMN 2.0 based variability approaches. In Section III we present the BPMNext language and in Section IV we present how BPMNext is used in a real case study, as well as its tool support. Finally, in Section V we present some conclusions and an outline of future work.

## II. APPROACHES FOR MODELING PROCESS FAMILIES

In what follows we present the results of a study of approaches for modeling process families. The study is focused on BPMN 2.0 based approaches, as well as language independent approaches that were adapted to BPMN 2.0.

In [4], [5] the authors performed comprehensive surveys about variability modeling including other complimentary approaches. However, they did not pay special attention on BPMN-based approaches. In fact, they have ignored some of the approaches that will be presented next.

For performing a practical comparison, we modeled in each language an airport check-in process taken from [9]. In such paper, the authors use this case study as a practical example within the VIVACE framework for the systematic evaluation of variability support in process-aware information systems.

#### A. Summary of approaches

PROcess Variants by OPtions (Provop, [10]) is a language independent variability by restriction/extension approach. The language allows the definition of configurable regions and a set of configuration alternatives together with context rules that restrict the application of such alternatives. These rules support the definition of dependencies between alternatives with respect to their joint existence or mutual exclusion, their order of execution (e.g., an alternative cannot be executed before another one), and a hierarchical relation (i.e., a combination of a dependency and an order of execution rule). This approach provides a very simple, modular and expressive mechanism for variability modeling. However, the language only admits variation points with respect to activities (tasks and subprocesses). Moreover, the modeling tool is not currently available. This approach is related to our work since it was adapted for BPMN 2.0 in [11], [12], [13].

The Common Variability Language (CVL, [14]) is a language independent variability by extension approach. However, in [14] the authors study its applicability to process families based on the use of BPMN. They define how CVL and BPMN can be used together for expressing variation points on activities, besides the language can potentially involve other elements. The extension mechanism separates the specification of the base model (in this case a process model) and its variants (called placed fragments), achieving better readability, greater comprehension and scalability. The approach also uses a variation model defining which replacement fragments are used in each variation point, and a resolution model which specifies contextual conditions that must apply for configuring a model. The approach is supported by available tools.

vBPMN [15] is another variability by extension approach. The approach combines BPMN 2.0 models with the rule language R2ML for managing variabilities on activities, gateways and events at runtime (what is called flexibility-by-design [16]). The customizable process is built together with a data context and adaptive workflow segments which depends on the context. The segments can be any structured block of the model, i.e., with only one entry flow and only one exit flow, which are decorated with additional graphical elements. The approach also defines event-aware adaptation patterns which take place when entering a segment and define the available variants which range from simple behaviors (e.g., skipping of tasks) to complex interrelated behaviors between variants. The language R2ML provides a way of connecting the context data

with the adaptation patterns, and such rules are evaluated at runtime. The adaptation patterns are hard to read and the use of the rules language adds complexity on the definition of the process family. Finally, there is no supporting tool available.

Process Family Engineering in Service-Oriented Applications (PESOA [17]) project provides a language independent variability by restriction approach which defines the notion of a variant-rich process model, i.e., a process model that contains every possible variant of the process family. Activities and data within the model can be annotated to be optional, replaceable by another element, conditioned to some constraint, and so on. The overload of annotations together with the need of expressing every variant in the same process, affects the understandability of the process family. The approach also uses a feature model which determines the context conditions to which each variant applies. However, unlike other approaches, e.g., Provop, it is not possible to state relations between variants of different variation points. The approach is supported by an available tool, only providing modeling capabilities, not supporting the customization process. Although the approach is language independent, it is also related to our work since it was adapted for BPMN 2.0 in [18], [19].

Configurable-BPMN (C-BPMN [20]) follows the ideas of the Configurable Event-driven Process Chain (C-EPC, [21]) approach, providing a variability by restriction approach focused on BPMN 2.0. It defines configurable fragments which are structured blocks (Simple Entry, Simple Exit fragments). These fragments can be expressed within a pair of configurable connector followed by a set of flows representing the alternative variants. This is similar to the use of a OR gateway expressing different possible flows of execution according to some inclusive conditions. The fragments are equipped with expressions that define context conditions, e.g., the inclusion, exclusion, and optionality of flows, and relations between separate fragments. Process customization is achieved by using a questionnaire that express a specific context configuration. In this case, the variation point can be expressed on activities and gateways marking them with a thick edge. There are also expressions that allow activities to be activated, suppressed or defined as optional, and gateways to be kept, deleted or restricted to some specific sequence flow. It is a very expressive approach since it allows variation point on activities, data, roles and gateways, but EPC is less expressive than BPMN 2.0. Unlike C-EPC, it does not provide any supporting tool.

BPMNt [22] is a conservative extension of BPMN 2.0. i.e., not affecting its base semantics, providing a variability by extension approach. It extends BPMN 2.0 metamodel for introducing elements capturing the semantics of deleting, replacing and adding any kind of process element. Despite its great potential, it has a poor graphical representation which affects understandability. The approach proposes to build tree-based representations of a process model representing a specific customizing configuration. For each variation point represented in the tree, i.e., a process element, it is possible to express deletion, replacement and adding of variants, as well as relations between them. Tool support is not available.

TABLE I  
COMPARISON OF DIFFERENT APPROACHES

Approach	Variability kind	Representation	Variation points	Language	Tool
Provop [11], [12], [13]	Both	Base model + Context model	Activities	Language independent	✗
CVL [14]	Extension	Base model + Context model	Activities	Language independent	✓
vBPMN [15]	Extension	Base model + Context model	Activities, gateways, events	BPMN 2.0 extension	✗
PESOA [18], [19]	Restriction	Base model + Feature model	Activities, data	Language independent	✓
C-BPMN [20]	Restriction	Integrated model	Activities, gateways	BPMN 2.0 extension	✗
BPMNt [22]	Extension	Tree-based representation	Activities, data, roles, gateways, events	BPMN 2.0 extension	✗
BPMN* [23]	Restriction	Base model + Feature model	Activities, data, roles, gateways, events	BPMN 2.0 extension	✗

Finally, BPMN\* [23] is BPMN 2.0 extension providing a variability by restriction approach. It consists of adding stereotypes and tags to the process elements (activities, data, roles, gateways, and events), which are related with a feature model. It extends the BPMN 2.0 metamodel for expressing the relation between variation points and their respective variants. It is a simple approach with great expressiveness. As in the case of the other variability by restriction approaches, it has a scalability problem related to the specification of alternatives in the same model. Tool support is not available.

In Table I there is a summary of the approaches, in particular: how the variability is expressed, the models required for the specification, the kind of elements over which the variability is expressed, the focused language and tool support.

Focusing on BPMN 2.0 based approaches, we can see that they do not provide any available supporting tool. vBPMN is focused on providing flexibility-by-design but it is the harder to read because of the use of a rules language. C-BPMN is the simpler language of all of them. BPMNt is a very powerful approach but it does not define how the variation points and variants can be expressed using process models. Finally, BPMN\* is one of the most expressive approaches but the inclusion of variants within the customizable process affects understandability.

In general terms, the variability by extension approaches require the definition of more components than in the case of variability by restriction approaches. However, they are more scalable in terms or the understandability of the models when the process family increases.

### B. Classification of approaches

As a final conclusion about this study we can classify the different approaches with respect to how they represent process families.

1) *Provop*, *CVL*, *vBPMN*: They are variability by extension approaches based on a customizable process, a context model and some way of expressing customizing rules. They are focused on variation point on activities and in general their models are simple to understand. Provop also provides variability by restriction.

2) *PESOA*, *BPMN\**: Besides one of them is a general approach and the other is an extension of BPMN 2.0, both are variability by restriction approaches and share the definition

of stereotypes for expressing variation points. They both use a feature model for expressing possible configurations.

3) *C-EPC*, *C-BPMN*: They are also variability by restriction approaches. However, they do not use stereotypes but tags for expressing variation points. They are a little bit more invasive with respect to the customizable process, in comparison to the other approaches. This means that it is hard to follow what they try to represent.

4) *BPMNt*: This is a completely different approach in the sense that it has no process-based representation of variation points and variants.

### C. vSPEM

As a complement of this work we also looked at vSPEM [7], [8], which is an extension of the Software & Systems Process Engineering Metamodel Specification (SPEM, [6] for modeling software processes, that could be taken into account for BPMN variability. There is a specific line of work for software process families based on SPEM where variants of the process are obtained by tailoring the core process.

SPEM defines specific elements for software process modeling such as disciplines, tasks, activities, roles, work products, and processes. The method content view allows defining such elements and the process view instantiates these elements in a specific process. It also provides a way for modeling variability, defining several variability types between two related elements, such as: replaces, extends, extends-replaces, contributes and not assigned.

However, the language has some limitations for modeling software process families, e.g., it is difficult to visualize the core process which contains the common elements of the process family. Taking into account the limitations, the variability extension for SPEM, named vSPEM, provides the means for a direct specification of the variability in a process.

In vSPEM, variation points are defined within the process, along with the variants that can be selected to fill the points. The relation between a variation point and a variant is a *replaces* relation from SPEM, and when performing process tailoring each variation point is substituted with exactly one variant. As examples of use, a variation point can have one or more associated variants, so when a process is tailored one variant will be selected to fill the variation point. Another case is when a variation point have only one associated variant, so, in the tailored process the variant can be selected or not.













	Activity	WorkProductUse	RoleUse	TaskUse
Base Element				
VarPoint				
Variant				

Fig. 1. Main v-SPEM concepts

v-SPEM extends the SPEM metamodel with new elements and defines a new notation to represent them, as presented in Figure 1.

### III. INTRODUCING BPMNEXT

In this section we present the BPMN extension we propose to model variability in business process families, discussing previously the selection of the base approach we have used.

#### A. Discussion of existing approaches

As presented in Section II the variability by extension approach seems to be easier to understand for medium size and large scale process families, but variability by restriction presents all possible variants within the base process showing the options explicitly thus providing a complete view of the family and its variants.

Regarding the languages used to model variability, we found that although BPMN 2.0 is nowadays one of the most used languages for business process modeling, there are not many approaches that provide complete support for it, including modeling and configuration facilities. What is more, we can see that they do not provide any available supporting tool for BPMN 2.0 variability.

Although Provop, PESOA and C-EPC are well-known approaches that provide support for the core elements of process variability, as shown in the evaluation carried out within the VIVACE framework [9], they do not provide practical elements for BPMN 2.0. Although the first two can be used within BPMN 2.0 models, there are no tool support for applying them.

We believe that showing variation points and the corresponding variants for each one provides a better view of the family, but we also want to minimize the complexity of adding all elements to the base process. Based on the analysis of existing approaches and trying to provide both conceptual and tool support for BPMN 2.0 variability by restriction, we decided to translate the vSPEM approach for software processes variability, to business processes variability, thus defining BPMNext, a BPMN 2.0 extension supporting business process families.

We basically take two aspects of vSPEM: how variation points are represented, and how the BPMN 2.0 metamodel is extended. As with vSPEM, in our approach the relation

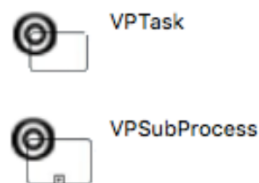


Fig. 2. BPMNext variation points in activities

between a variation point and a variant is a replacement relation, and for process customization, each variation point is substituted with exactly one variant. Other software process aspects that vSPEM handles naturally, e.g., work product, make no sense in our proposal and, thus, they were discarded.

#### B. Introducing BPMNext

In what follows we define how variation points and variants are defined, as well as the set of constraints that apply for process customization.

We restrict BPMNext to variation points on activities (tasks and subprocesses) and roles (lanes). In the case of activities, we tried to make a proposal as generic as possible. In particular, we can substitute an activity with any other activity, no matter if the task is of some specific type, e.g., manual, service, script, etc., or if it is another subprocess. Activities deletion is also possible, only for activities with a simple entry and a simple exit flow.

In this case, process customization involves the connection of the precedent and subsequent flows. We also consider the possibility of changing the role in charge of some activity, introducing new roles, renaming them, or keeping them in the same place. Finally, a very interesting feature of the language is that it allows variants also with variation point, allowing many levels of expressiveness. This is something that it is not present in the other approaches and it is not clear, but feasible, in vSPEM.

We decided to express variation points in a simple and clear way, and thus we extend BPMN 2.0 with two graphical elements corresponding to tasks and subprocesses. This is similar to the way vSPEM specifies variation points and, as depicted in Figure 2, it consists of the standard representation of activities together with a double circle in the upper left side of the element.

A process family requires the modeling of the base customizable process together with the variants as independent process models. These variants can also be customizable processes with their own variants. In this way we allow a multi-level modeling of variants.

#### C. Extending the BPMN metamodel

We have extended the BPMN 2.0 metamodel as depicted in Figure 3, to include variability concepts.

We defined a concept VELEMENT which is the base class of the extension. Variation point (in pink) are an extension of the corresponding BPMN 2.0 elements (in blue). In particular, we

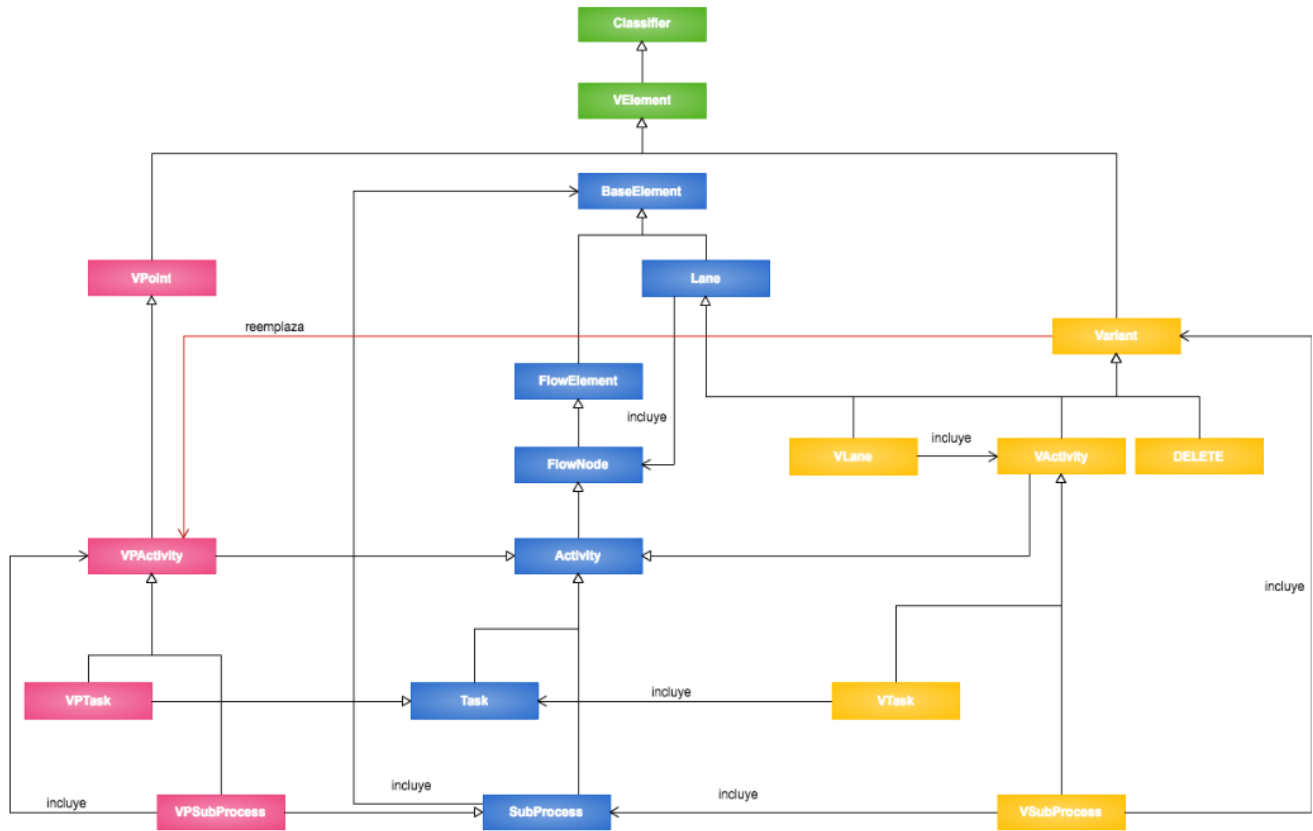


Fig. 3. BPMNext metamodel (excerpt)

defined VPTask and VPSubprocess which are extensions of the Task and SubProcess elements, respectively. a VPSubProcess is composed of other VPActivity allowing multi-level variation points.

Variants (in yellow) are also extensions of the corresponding BPMN 2.0 elements. A Variant is related to a general VPActivity allowing the kind of substitution we have described before: a VPActivity can be replaced by a task (VTask) or a subprocess (VSubProcess), it can be performed by another role (VLane) or it can be deleted (DELETE).

This metamodel can be easily extended adding other BPMN 2.0 elements as variation points, e.g., gateways. However, it is needed to define first the way variation points can be graphically expressed in the process model.

#### IV. BPMNEXT IN PRACTICE

In [24] we presented an experience report from a process management improvement initiative we have carried out within our university. We have worked with the General Direction of Cooperation and International Relations (Dirección General de Cooperación y Relaciones Internacionales, DGRC), regarding academic exchange programs for graduate and postgraduate students and teachers, and collaboration projects between the university and external institutions (national and international).

We modeled the "Assignment of positions in academic exchange programs" BP, in which for every exchange pro-

gram in which the university participates (Erasmus, Santander, Fulbright, etc.), positions are announced and interested people (students, faculty, staff) apply for them. The DGRC and other authorities analyzed applicant's merits and after some steps and meetings define the assignments, which are then notified to the applicants. In Figure 4 we present its model using BPMN 2.0, without expressing other participants and the corresponding interactions.

We clearly identified a process family context where all programs share a common set of activities, and based on specific conditions of each program (variation points) different paths are executed, probably involving different participants. In such opportunity, we decided to use only BPMN 2.0, and to define blocks based on XOR gateways that will be executed or not depending on the programs settings.

We selected this option since it was the easiest way we find to immediately solve the problem at hand and also to provide a simpler way for business people to understand the modeling of all exchange programs within a unique model. To do so for each program we present the corresponding variant with its execution path colored so they can visualize for each program which is the corresponding control flow.

In terms of process customization, we implemented a prototype using Bonita BPMS. We used specific tables for setting and loading the configuration of each exchange program for



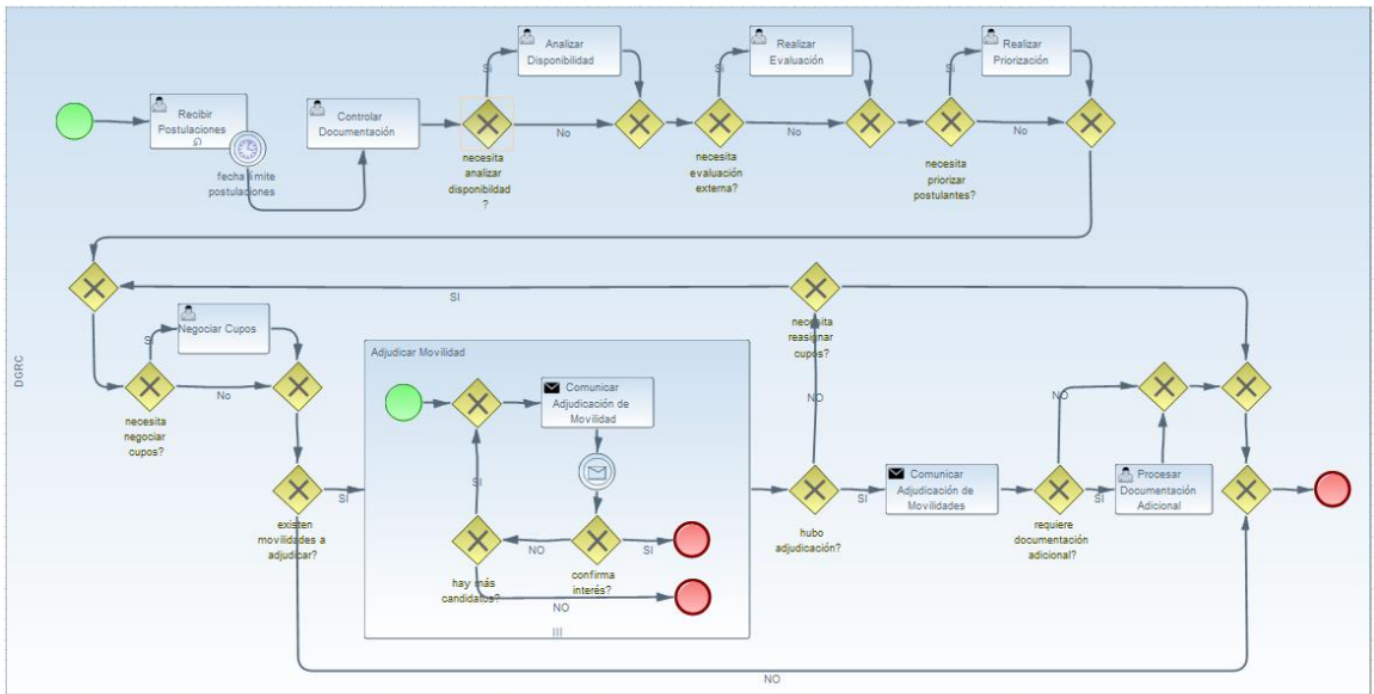


Fig. 4. "Assignment of positions in academic exchange programs" BPMN 2.0 original model

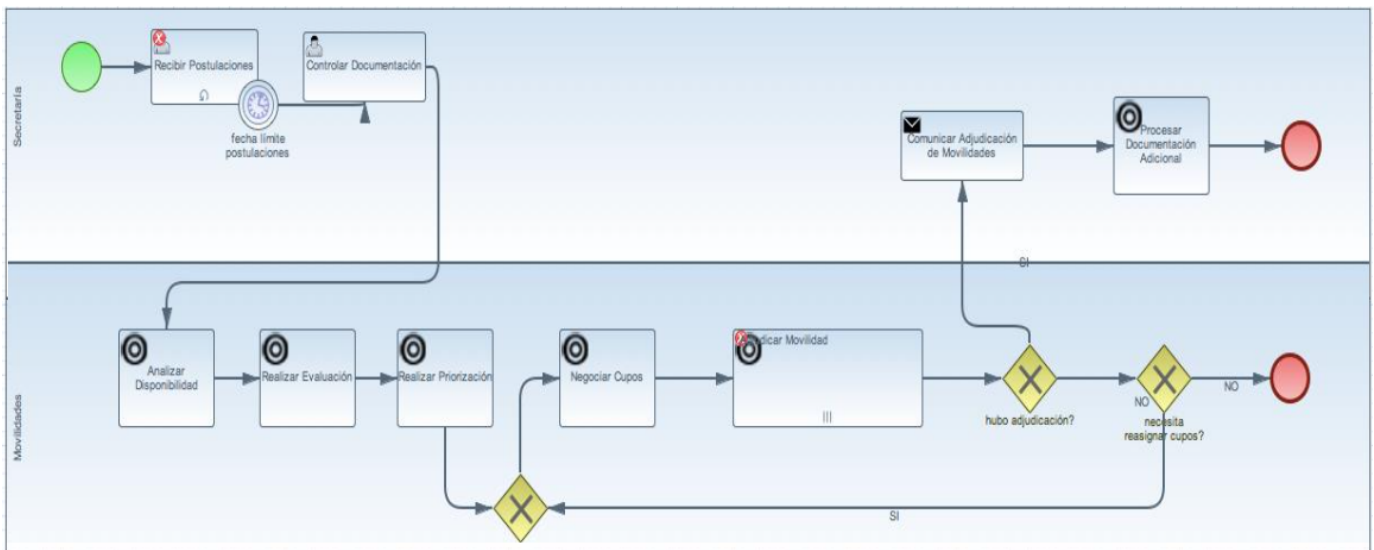


Fig. 5. "Assignment of positions in academic exchange programs" BPMNext model

the execution of defined variants. When executing the BP, the first selection the user makes is the exchange program, and based on that process variable, the variables of each XOR condition are set from the data in the database.

In what follows we present another approach for the modeling and customization of this process family, based on the use of BPMNext and a tool within the Eclipse framework.

#### A. Modeling with BPMNext

We use BPMNext in order to handle variation points. The language is supported in practice by an extension of the

Eclipse BPMN2 Modeler. The process, depicted in Figure 5 is simpler than the original, since it requires less gateways to express the control flow and it allows to visually detect whether the variation points are. Moreover, except for the case of "Adjudicar Movilidad", which is a subprocess, the other variation points can be represented as task variation points, i.e., in which they can be replaced by a simple task.

#### B. Process customization

For customizing the base process, we also extended the Eclipse BPMN2 Modeler. We have implemented a wizard



In this paper, we have presented the results of a detailed study about the modeling and customization of business process families based on BPMN 2.0. We also considered other approaches in order to extend the study and compare their characteristics. We have classified the proposals in four categories with respect to the way they handle the variability. As a general conclusion, the variability by restriction approaches are more complex but also more used since they provide a complete view of the process family. Moreover, there is little tool support for BPMN 2.0 based approaches.

As a complement of this study, we essayed the definition of a novel approach based on the ideas provided by vSPEM, a language for the modeling of software processes families which extends the SPEM language. We introduced BPMNext, an extension of BPMN 2.0 that supports the variability by restriction approach, translating the definitions in vSPEM to BPMN 2.0. The language considers variation points with respect to activities (tasks and subprocesses) and roles.

The customizable process and the variants are independently modeled in such a way that the process customization and the maintenance of the variants are simplified. The adoption of the way vSPEM extends the SPEM metamodel, provides a very simple way of extending BPMNext with the consideration of variation points in other elements, e.g., gateways and events. However, this extension requires future work in order to precisely define the meaning of a variation point and the study of consistency aspects that process customization requires in such cases. Also, we will analyze the native BPMN extension mechanism to support the extension.

We also built a supporting tool as an extension of the Eclipse BPMN2 Modeler. The tool allows not only the modeling of the customizable process and their variants, but also the process customization. One of the most interesting aspect for both the modeling extension and the customization process is that the approach supports multilevel variability, i.e., a variant can also contain variation points with their respective variants. There are many open issues that can be considered as future work, e.g., the definition of better error messages, and the support of collaborative processes (i.e., many pools), among others.

The use of BPMNext in a real case study about the assignment of positions in academic exchange programs showed the feasibility of the proposal. The language allowed to represent the original problem in a simpler way improving its understanding. In order to analyze its real potential, it is desirable a more specific comparative evaluation with the rest of the approaches.

Finally, we want to essay a model-driven approach for carrying out the customization process instead of using a programming language as it is now. This could provide a higher-level representation of the transformation steps involved in the customization process and the constraints that must apply between the alternatives. It will also provide modularization, reuse of knowledge and improved maintainability for the automated generation of process variants.

This work was partially funded by the Comisión Sectorial de Investigación Científica (CSIC), Universidad de la República, Uruguay. We would like to thank the undergraduate students who worked in the project: Ignacio Betancurt, Alejandro Brusco and Nicolás Dinetti.

## REFERENCES

- [1] M. Weske, *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer, 2012.
- [2] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, "Business process management: A survey," in *Business Process Management, International Conference, BPM 2003, Eindhoven, The Netherlands, June 26-27, 2003, Proceedings*, ser. Lecture Notes in Computer Science, W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, Eds., vol. 2678. Springer, 2003, pp. 1–12.
- [3] OMG, "Business Process Model and notation (BPMN) v2.0," Tech. Rep., 2013.
- [4] M. L. Rosa, W. M. P. van der Aalst, M. Dumas, and F. Milani, "Business process variability modeling: A survey," *ACM Comput. Surv.*, vol. 50, no. 1, pp. 2:1–2:45, 2017.
- [5] G. Valenca, C. Alves, V. Alves, and N. Niu, "A systematic mapping study on business process variability," *International Journal of Computer Science & Information Technology (IJCSIT)*, vol. 5, no. 1, 2013.
- [6] OMG, "Software and Systems Process Engineering Metamodel (SPEM) v2.0," Tech. Rep., 2008.
- [7] T. Martínez-Ruiz, F. García, and M. Piattini, *Towards a SPEM v2.0 Extension to Define Process Lines Variability Mechanisms*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 115–130.
- [8] T. Martínez-Ruiz, F. García, M. Piattini, and J. Munch, "Modelling software process variability: an empirical study," *IET Software*, vol. 5, no. 2, pp. 172–187, 2011.
- [9] C. Ayora, V. Torres, B. Weber, M. Reichert, and V. Pelechano, "VIVACE: A framework for the systematic evaluation of variability support in process-aware information systems," *Information & Software Technology*, vol. 57, pp. 248–276, 2015.
- [10] M. Reichert, S. Rechtenbach, A. Hallerbach, and T. Bauer, "Extending a business process modeling tool with process configuration facilities: The provop demonstrator," in *Proceedings of the Business Process Management Demonstration Track (BPMDemos 2009)*, ser. CEUR Workshop Proceedings, vol. 489. CEUR-WS.org, 2009.
- [11] E. Santos, J. Pimentel, J. Castro, J. Sánchez, and O. Pastor, "Configuring the variability of business process models using non-functional requirements," in *Enterprise, Business-Process and Information Systems Modeling - 11th International Workshop, BPMDS 2010, and 15th International Conference, EMMSAD 2010, Proceedings*, ser. Lecture Notes in Business Information Processing, vol. 50. Springer, 2010, pp. 274–286.
- [12] I. Machado, R. Bonifácio, V. Alves, L. Turnes, and G. Machado, "Managing variability in business processes: An aspect-oriented approach," in *Proceedings of the 2011 International Workshop on Early Aspects*, ser. EA '11. ACM, 2011, pp. 25–30.
- [13] T. Nguyen, A. Colman, and J. Han, "Modeling and managing variability in process-based service compositions," in *Proceedings of the 9th International Conference on Service-Oriented Computing*, ser. ICSOC'11. Springer, 2011, pp. 404–420.
- [14] C. Ayora, V. Torres, V. Pelechano, and G. H. Alférez, "Applying cvl to business process variability management," in *Proceedings of the VARIability for You Workshop: Variability Modeling Made Useful for Everyone*, ser. VARY '12. ACM, 2012, pp. 26–31.
- [15] M. Döhning and B. Zimmermann, "vbpmm: Event-aware workflow variants by weaving BPMN2 and business rules," in *Enterprise, Business-Process and Information Systems Modeling - 12th International Conference, BPMDS 2011, Proceedings*, ser. Lecture Notes in Business Information Processing, T. A. Halpin, S. Nurcan, J. Krogstie, P. Soffer, E. Proper, R. Schmidt, and I. Bider, Eds., vol. 81. Springer, 2011, pp. 332–341.
- [16] M. Reichert and B. Weber, *Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies*. Springer, 2012.



- [17] F. Puhlmann, A. Schnieders, J. Weiland, and M. Weske, "Variability mechanisms for process models," PESOA-Report TR 17/2005, Process Family Engineering in Service-Oriented Applications (PESOA). BMBF-Project, Tech. Rep., 2005.
- [18] V. Kulkarni and S. Barat, "Business process families using model-driven techniques," *IJBPM*, vol. 5, no. 3, pp. 204–217, 2011.
- [19] G. Gröner, M. Boskovic, F. S. Parreiras, and D. Gasevic, "Modeling and validation of business process families," *Inf. Syst.*, vol. 38, no. 5, pp. 709–726, 2013.
- [20] H. Zhang, W. Han, and C. Ouyang, "Extending BPMN for configurable process modeling," in *Proceedings of the 21st ISPE Inc. International Conference on Concurrent Engineering, Beijing Jiaotong University, China, September 8-11, 2014*, ser. Advances in Transdisciplinary Engineering, vol. 1. IOS Press, 2014, pp. 317–330.
- [21] M. Rosemann and W. M. P. van der Aalst, "A configurable reference modelling language," *Inf. Syst.*, vol. 32, no. 1, pp. 1–23, 2007. [Online]. Available: <http://dx.doi.org/10.1016/j.is.2005.05.003>
- [22] R. M. Pillat, T. C. Oliveira, P. S. C. Alencar, and D. D. Cowan, "Bpmnt: A BPMN extension for specifying software process tailoring," *Information & Software Technology*, vol. 57, pp. 95–115, 2015.
- [23] M. Terenciani, D. M. B. Paiva, G. Landre, and M. I. Cagnin, "Bpmn\* - A notation for representation of variability in business process towards supporting business process line modeling," in *The 27th International Conference on Software Engineering and Knowledge Engineering, SEKE*, H. Xu, Ed. KSI Research Inc. and Knowledge Systems Institute Graduate School, 2015, pp. 227–230.
- [24] A. Delgado and D. Calegari, "Changing the focus of an organization: From information systems to process aware information systems," in *Enterprise, Business-Process and Information Systems Modeling - 16th Int. Conference, BPMDS, Proceedings*, ser. Lecture Notes in Business Information Processing, vol. 214. Springer, 2015, pp. 53–67.