UNIVERSIDAD POLITECNICA DE VALENCIA

PROS
Centro de Investigación en Métodos de Producción de Software

Informe Técnico / Technical Report




Variability Management in Business Process Models

Clara Ayora, Victoria Torres, Vicente Pelechano





Ref. #:	ProS-TR-2012-17				
Title:	Variability Management in Business Process Models				
Author (s):	Clara Ayora, Victoria Torres, Vicente Pelechano				
Corresponding author (s):	cayora@pros.upv.es vtorres@pros.upv.es pele@pros.upv.es				
Document version number:	1.0	Final version:	Yes	Pages:	15
Release date:	December 2011				
Key words:	Business Process Modeling, Variability Modeling, Variability Management				

Universidad Politécnica de Valencia · Camino de Vera s/n · Edificio 1F · 46022 Valencia Spain ·
T. +34 96 387 70 07 Ext. 83530 · M. + 34 619 543 623 · F. +34 96 387 73 59 · info@pros.upv.es · www.pros.upv.es

Variability Management in Business Process Models*

Clara Ayora, Victoria Torres, and Vicente Pelechano

Centro de Investigación en Métodos de Producción de Software
Universitat Politècnica de València
Camino de Vera s/n, 46022 Valencia, Spain
{cayora,vtorres,pele}@pros.upv.es

Abstract. Business Process (BP) models capture the coordination of activities whose execution realizes business goals. However, their construction entails big challenges, especially when BPs exist in many variants. In these cases, variability modeling techniques become essential to succeed in BP modeling. Contemporary BP variability modeling approaches lack of variant management and/or variability expressiveness. Either the variability is difficult to maintain or it is not fully covered over the modeling elements. In this work, we propose an approach that deals with BP variability modeling. The approach advocates for separating variability from the BP model. Thus, variability is handled independently facilitating its management and maintenance. The approach also provides techniques to model variability over the whole set of modeling elements, which enhances variability expressiveness. For such purpose, we rely on Software Product Lines techniques, which allow promoting model legibility and reuse regarding variability. A case study illustrates the applicability of the approach.

Keywords: Business Process Modeling, Variability Modeling, Variability Management

1 Introduction

Business Process (BP) models represent, by means of tasks and resources, how organizational goals are achieved. These models are commonly built for organization and/or information system design purposes. In both cases, to take advantage as much as possible of these models, they should represent in an accurate way the organizational reality that is interesting for a specific goal [1]. However, BP modeling is not a trivial task, being usually highly dependent on the modeler skills [2]. In addition, when the BP being modeled contains high level of variability[1], the complexity of the modeling process increases, turning models into artifacts that are error-prone and complex to build, manage and understand.

* This work has been developed with the support of MICINN under the project EVERYWARE TIN2010-18011.

[1] i.e. when the same BP is presented in different variants for different contexts

Coping with such variability modeling issues constitutes one of the challenges currently faced by the BPM research area. Good examples of such interest are the approaches developed by *Rosemann et al.* [3], *Puhlmann et al.* [4], or *Hallerbach et al.* [5]. These approaches face the modeling challenges that arise when dealing with BP variability. However, even though they constitute a solid base to face such challenges, we still find some limitations. In particular we find lacks when (1) managing (evolving or modifying) BP variants and (2) dealing with variability in several modeling elements (e.g. control flow, roles, documents, events, etc.). To provide proper support to these two challenges turns crucial to successfully model BP variability.

In order to address these lacks, in this work we rely on the Software Product Line (SPL) research area. The techniques developed in SPL look for creation of a portfolio of similar software systems by avoiding duplication and making explicit the rationale for variations [6]. In addition, SPL techniques have been put into practice to deal with variability in other research areas (e.g. autonomic computing [7]) because of their simplicity and reuse capabilities [8]. In this work we use some of these techniques within the context of Business Process Modeling Languages (BPMLs). Specifically, we take and adapt the Base-Variability-Resolution (BVR) approach [9], where variability is represented explicitly and independently of the variants' commonalities. To implement such approach we make use of *Common Variability Language* (CVL, the new standard being developed for variability modeling) [10, 11] and *feature models* [8]. As a result, the proposed approach improves the management of BP variants and it provides support to deal with variations regarding any of the elements included in a BP model (roles, objects, events, tasks, control flow, etc.).

The remainder of the paper is structured as follows. Section 2 introduces the case study that exemplifies the approach. Then, Section 3 introduces the SPL background in which this work relies on. Section 4 explains how SPL techniques are used in this work to deal with BP variability modeling. Section 5 proposes a methodology to guide modelers when applying the proposed approach. Next, Section 6 provides an overview of the contemporary BP variability modeling approaches and discuss about how they support BP variability modeling. Finally, Section 7 concludes the work and outlines future research directions.

2 Case study

To exemplify the applicability of the proposed approach we make use of the minor contract process used by the Valencian Regional Ministry to manage the contracts whose cost is less than 30.000€. The process presents high degree of variability regarding control-flow and the involved roles and documents. The process is made up mainly of four sub-processes (beginning of the procedure, contract award, contract execution, and contract settlement) which vary depending on the type of contract being required. The process starts by the *Petitioner department* when a contract is required. From this point, the rest of the process depends on the type of the required contract. If the contract refers to Information Technologies (IT), then it is handled by the *Central dealing service*, which

is common to all the departments of the Regional Ministry. On the contrary, the contract is handled directly by the *Dealing unit* of each department. Once the procedure has started, the contract needs to be awarded. The resulting documentation depends as well on the required contract type. Contracts of IT are different than contracts of Supplies or Public Works since the required information and the involved materials are different. Afterwards the contract has being awarded, it is executed. During this sub-process, it is necessary to register the documents resulted in the contract award. This registration could be done either on-line, if the *dealing service* has the proper technology, or manually if it does not have it. In order to control the manual registration, the manager of the corresponding *dealing service* should be notified at the same time that the registration is being performed. After, the next step is to prepare the necessary stock list with the materials needed for the contract. Then, a budget is prepared to cover the expenses of this list. Once it is prepared, the budget needs to be arranged. This arrangement can be done again on-line or manually, depending on the technology of the *dealing service* involved. Finally, the process finishes by settling the contract.

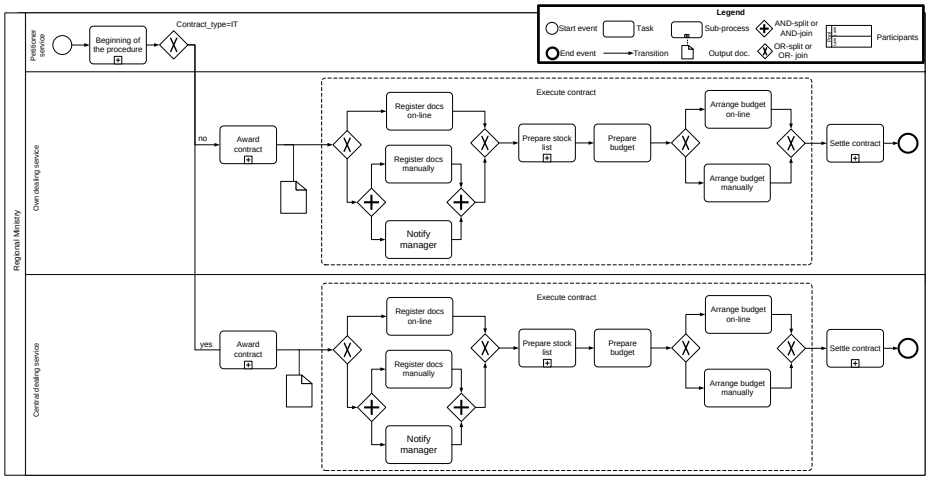

Fig. 1. Minor contract process model including all process variants

Figure 1 shows the minor contract process model in *Business Process Modeling Notation* (BPMN), with all process variants represented together in the same model. It depicts how the process is carried out by either the *dealing unit* or by the *Central dealing service* depending on the contract type. Thus, process variants depend on (1) the required contract type and on (2) the technology that the involved department has. For instance, Figure 2 shows the process variant regarding the *Justice and Social Welfare* (JSW) department, where no technology is available neither for document registration nor budget arrangement.

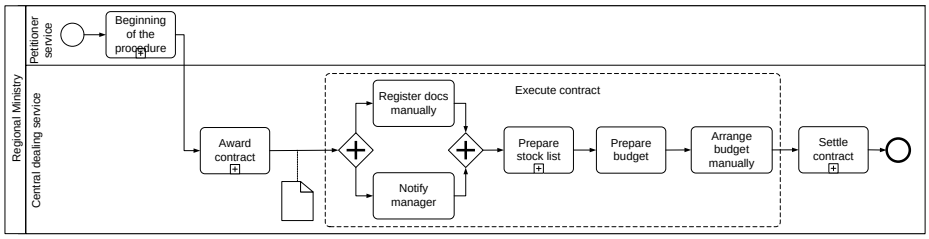

Fig. 2. Minor contract process model for *Justice and Social Welfare* department

3 Background and foundations

Initially, BP variability modeling approaches attempted to model variability only in one model (e.g. reference process models). As it is discussed in [12], using conditional branching to represent together all process variants into one single model results in huge models, which are less scalable and difficult to understand and maintain. Besides, models are cluttered with variability specification where variant particularities are mixed up with normal branching. This do not allow modelers to distinguish if the branching conditions are variant-specific or part of the common logic. On the contrary, if variability is modeled separately, it could be managed independently reducing its impact over the variants' commonalities [9]. Thus, variability is brought as first-class aspect of the modeling process.

For the purpose of modeling separately BP variability, we propose to transfer the ideas and techniques from the SPL area. SPL is focused on the development of techniques for the creation of a portfolio of similar software systems. The objectives of SPL are: *commonality capitalization* and *variation management* [6]. On the one hand, *commonality capitalization* avoids duplication and divergence of shared assets. On the other hand, *variation management* allows making explicit location, rationale and dependences for variations. Addressing properly these aspects allows reducing time, effort, cost and complexity of creating and maintaining a family of similar software systems [6]. Thus, by using SPL ideas we are gaining in model reusability and scalability. Specifically, we make use of the Base-Variation-Resolution (BVR) modeling approach and Feature Modeling to build BP models with variability. Figure 3 shows both techniques.

3.1 Base-Variation-Resolution approach

BVR is widely used to represent variability in an suitable and effective manner [9]. Concretely, BVR overcomes model construction in three parts: (1) variants' commonalities (represented in the *Base Model*), (2) variants' individualities (represented in the *Variation Model*), and (3) variants' configuration (represented in the *Resolution Model*).

Left side of Figure 3 shows the BVR approach. As it is depicted, the main advantage of BVR is that modelers are able to define, associated to one *Base Model* (e.g. minor contract process), as many *Variation Models* as they require.

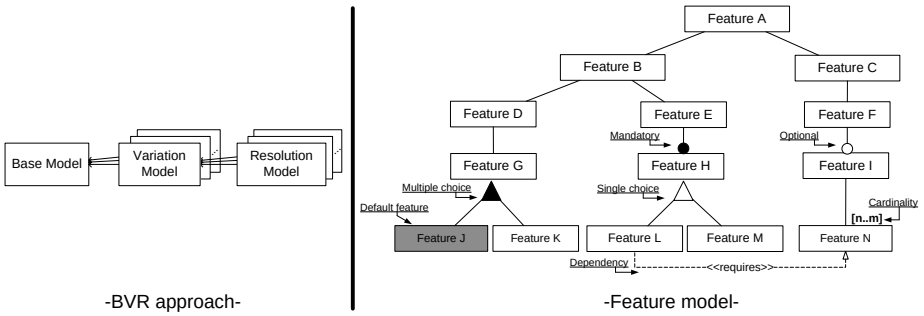

Fig. 3. SPL techniques

Ideally, modelers would build a different *Variation Model* for each domain where the *Base Model* is applied (e.g. *Justice and Social Welfare* or *Tourism*). Within each one of these *Variation Models* modelers then would define all the different alternatives that are possible in each domain. Therefore, this set of models would represent different variability scenarios at different levels regarding a common shared *Base Model*. Finally, the *Resolution Model* is where modelers specify which alternatives, from those defined in a specific *Variation Model*, constitute a valid model process variant for a specific domain. Therefore, there are as many *Resolution Models* as valid process variants exist. Thus, by using BVR, variability is represented by its own in an independent way.

3.2 Feature Modeling

Feature Modeling is a technique to specify the coarse-grained system functionality in terms of features [8]. These models have proven to be effective in hiding much of the complexity in variant specification definition [7]. Right side of Figure 3 depicts a *feature model*. As it is shown, features are hierarchically linked in a tree-like structure through variability relationships such as optional, mandatory, single-choice, and multiple-choice. More concretely, we make use of Czarnecki notation [13]. The concrete use we make of *feature models* is deeper presented and explained in the next section.

4 Base-Variation-Resolution approach for modeling business processes variability

Based on the SPL techniques presented previously, in this section we explain how we make use of them within the BP variability domain. BVR is a suitable approach to represent and manage variability. By defining a set of models, variability is managed independently, gaining in model reusability. Nevertheless, current BP models can be seen from a number of perspectives (including the resource perspectives and the documents), which can present alternatives too [14]. Trying to gather all these alternatives only in one model results in a scalability

problem [12]. To overcome this problem we propose a refinement of the BVR *Variation Model* based on the elements provided by BPMLs. This refinement, shown in Figure 4, results in two sub-models which refer to: (1) control flow and tasks (represented by the *Control Flow & Tasks Variation Model*) and (2) task related elements such as roles and objects (represented by the *Task Related Elements Variation Model*).

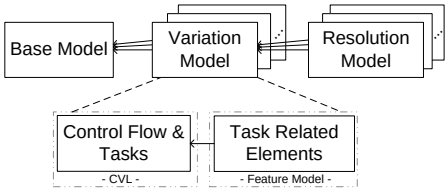

Fig. 4. BVR refinement

To implement BVR, we make use of *Common Variability Language* (CVL) for the *CFT-VM* and *feature models* for the *TRE-VM*. CVL [11] is a separated-language technique which advocates for the combination of a variability language (i.e. CVL) with other domain specific language (i.e. BPML). The two languages will each have their own concrete syntax, but the combination may still appear as an integrated concrete syntax. Thus, variability can be expressed separately and independently. In this work, we have taken BPMN for such combination, but it can be easily transferred into any other BPML since the CVL concepts are not tight to any notation. Regarding the *feature model*, we propose to structure it in five different levels. Each one of these levels is provided with a specific semantic that facilitates variability representation. For instance, the third level gathers for a specific task its variable related modeling elements (e.g. roles or documents).

Next subsections present a detailed explanation of the set of models that make up the complete approach. To facilitate the understanding and the potential of this approach we use the case study presented in Section 2.

4.1 Base Model

The *Base Model* is where modelers specify all the modeling elements of the BP shared by all process variants (i.e. variants' commonalities). In addition, modelers also define those model parts subject to vary (i.e. *variation points*, named *placements* in terms of CVL) where different alternatives can be applied depending on the specific context. These *placements* represent black boxes delimited by *boundary elements*, whose instantiation can be solved either at design time or at run time, depending when the alternative's selection is done. This differentiation is explicit by specializing the *placement* concept into *design time placements* (when the selection of an alternative depends on the initial context of the pro-

cess) and *run time placements* (when the selection of an alternative depends on the current context of a process instance).

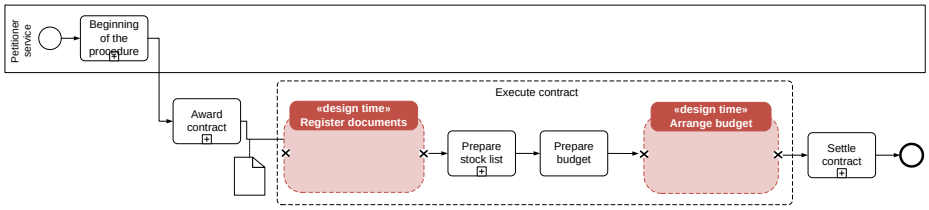

Fig. 5. Base Model for the minor contract process

Figure 5 depicts the *Base Model* of the case study. This model contains: (1) the set of modeling elements shared by all process variants (*start* and *finish* events, *Award contract*, *Prepare stock list* and *Settle contract* sub-processes and *Prepare budget* task) and (2) the set of *variation points* (*Register documents* and *Arrange budget placement*) whose instantiation depends on the available technology in the department. The different alternatives that fit in each *placement* are defined separately in the *Variation Model*.

Note that the role who carry out the last sub-processes can vary depending on the contract type. Since BPMN does not provide any technique to specify this type of variability, it cannot be represented in this model using such notation. This information will be specified later on in a separate model.

4.2 Variation Model

In contrast to the *Base Model*, the *Variation Model* gathers all the individualities introduced by each process variant, which can refer to any modeling element (e.g. tasks, control flow, roles, objects or events in BPMN). As we have already explained in this section, attempting to represent variability of each BP perspective only in one model results in a scalability problem. For this reason we propose to deal with it separately in two different models which are (1) the *Control Flow & Tasks Variation Model* and (2) the *Task Related Elements Variation Model*.

The Control Flow & Tasks Variation Model (CFT-VM) gathers, for each *placement* found in the *Base Model*, all the different alternatives (named *replacements* in terms of CVL) regarding process model control flow and tasks. These *replacements* are also delimited by *boundary elements*. Thus, when we want to instantiate a *replacement*, we only have to bind these *boundary elements* with the ones presented in the *placement* of the *Base Model* that is going to be solved. Figure 6 shows the graphical representation of the different *replacements*, for the *Register documents* and *Arrange budget placements*. Its instantiation will depend on the available technology in the involved department.

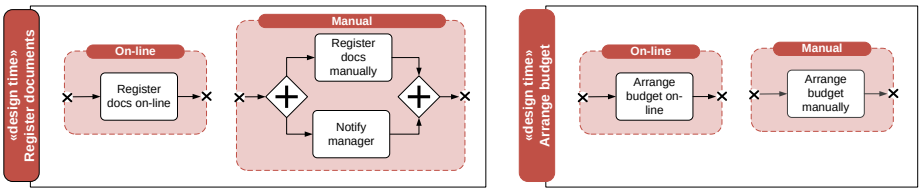

Fig. 6. The Control Flow & Tasks Variation Model for the minor contract process

In addition, there may exist semantic and structural dependencies (relationships) between *replacements* of different *placements*, e.g. the instantiation of a *replacement* excludes the instantiation of another *replacement* in another *placement*. For such purpose, associated to the *CFT-VM*, modelers can also build a *Control Flow & Tasks Constraint Model (CFT-CM)* to define and gather all these constraints. For instance, the type of relationships that can be considered are: *implication* and *mutual exclusion*. The *implication* relationship forces the selection of a *replacement* if another *replacement* (from a different *placement*) has been instantiated. On the contrary, the *mutual exclusion* relationship excludes the selection of a *replacement* if another *replacement* (from a different *placement*) has been instantiated.

The Task Related Elements Variation Model (TRE-VM) gathers the different alternatives that other BP perspectives (i.e. modeling elements beyond control flow and tasks) may have. Specifically, we propose the use of *feature models* to represent these alternatives. The main idea is to represent the alternatives as features associated to the sub-process/task/*placement* they affect. Thus, the features, and therefore the alternatives, can be selected or deselected depending on the corresponding process variant.

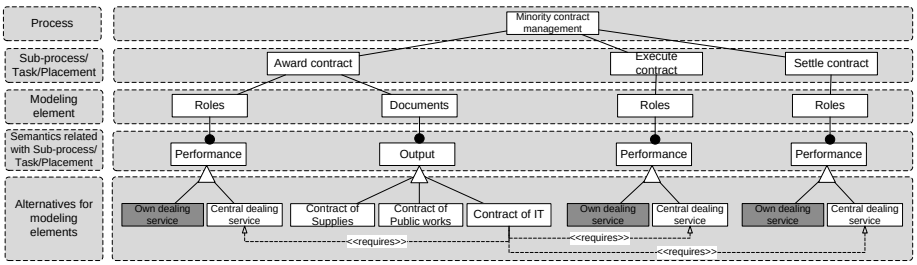

Fig. 7. The Task Related Elements Variation Model for the minor contract process

Figure 7 shows the TRE-VM for the case study. We structure this model in five different levels.

– The top level indicates the BP whose sub-processes/tasks/*placement* have variable related elements. In the example, the minor contract process.
– The second level indicates the sub-process/task/*placement* whose related elements are variable. In the figure, the *Award contract*, *Execute contract* and *Settle contract* sub-processes.
– The third level gathers the set of related variable modeling elements; in the example, we have included *roles* and *documents*. However, depending on the expressiveness of the used BPML, this level could be enlarged with many modeling elements as it is required (e.g. events) by simply adding new features in the tree-like structure.
– The fourth level specifies the semantics that relates the modeling elements defined in the third level with the sub-process/task/*placement* they affect. For instance, modelers could define the participation of different roles in the same task but behaving differently (e.g. differentiating between task performer and task supervisor).
– The fifth level specifies the alternatives that the modeling elements of the third level can have, but grouped by the semantics defined in the fourth level; For example, *Dealing unit* or *Central dealing service* are the roles who can carry out the sub-processes. In addition, constraints between different alternatives can also be defined by using "requires" or "excludes" constraints between features (e.g. if an IT contract is required, the performance of the sub-processes should be done by the *Central dealing service*).

4.3 Resolution Model

The previous set of models gathers, in conjunction, many BP process variants. Specifically, the *variation model* defines the different alternatives that can be applied to a specific *base model*. However, we still need to specify which combinations of these alternatives are valid for a particular context. This information is presented in the *resolution model*.

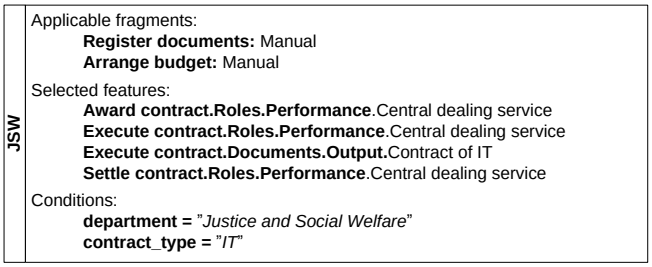

Fig. 8. Resolution Model for the *Justice and Social Welfare* department

Figure 8 illustrates the *Resolution Model* for the *Justice and Social Welfare* department. It is structured in three blocks. The first block (*Applicable fragments*) specifies, for each *design time placement* defined in the *Base Model*, the

selected *replacements* from the *CFT-VM* that constitute a specific process variant. For the case study, it is specified the *replacement* that is chosen for the *Register documents* and the *Arrange budget* of the *Base Model*. Note that not all the placements of the *Base Model* are considered in a resolution. Run time *placements* cannot be resolved before process deployment. However, if a *design time placement* is considered, just one *replacement* can be chosen for it.

Regarding the second block (*Selected features*), it gathers for each sub-process/task/*placement* whose related modeling elements are variable: (1) the appropriate features and (2) their cardinalities (if necessary). Specifically, for the example, it specifies that the *Central dealing service* is the role who carries out the sub-processes and the required contract is an IT contract.

Finally, the third block defines the conditions that determine the selection of these *replacements* and features based on the current context. Concretely, these conditions are specified by determining which context variables have to be evaluated to a specific value to define the process variant. For the case study, to obtain the process variant for the *Justice and Social Welfare* department with an IT contract, the context variable *department* has to be valued to "*Justice and Social Welfare*" and the *contract_type* variable to "*IT*".

Moreover, thanks to the definition of the *CFT-CM*, it is ensured the consistency and correctness of the specified resolutions. For instance, if a resolution which includes two *replacements* that are mutually excluded according to the *CFT-CM*, it should be forbidden and corrected.

5 Putting the approach into practice

Up until now, we have proposed an approach for BP variability modeling. Nevertheless, the construction of BP models constitutes a very complicated task. It requires therefore mechanisms to assist and guide modelers during this task [2]. In addition, when the model being constructed involves handling variability, as it is the case, the task becomes even more complicated. To reduce this complexity, Figure 9 presents the process that represents the way to proceed when putting into practice the proposed approach:

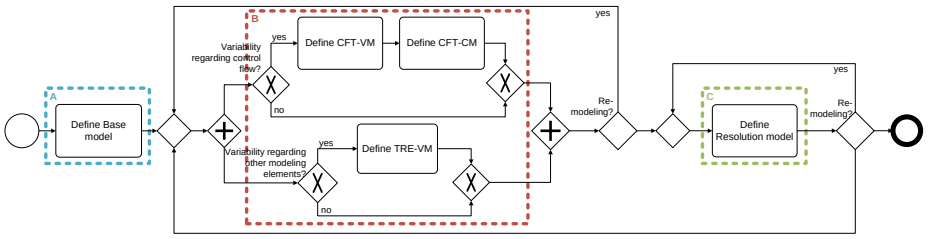

Fig. 9. Process to apply the approach

1. **Variants' commonalities specification stage** (Figure 9, mark A) The modeler should identify all variants' commonalities in the *Base Model* as well as which parts of the process are subject to variations (*placements*). It is also recommendable to identify at this early stage which context variables will determine the resolution of the *placements*. This will allow modelers to visualize from the very beginning which conditions make the process varying.
2. **Variants' individualities specification stage** (Figure 9, mark B) Modelers should identify the different alternatives that make up a specific process variant. If the alternatives refer to control flow structures and tasks, then the modeler should built a *CFT-VM*. For each *placement* identified in the model, the modeler should define all the different alternatives that fit in it. In turn, if the alternatives refer to other modeling elements such as roles or objects, then the modeler should build a *TRE-VM* for each subprocess/task/*placement* that present such alternatives. In the case of the *TRE-VM*, constraints about other modeling elements are defined explicitly in the associated feature diagram through the Czarnecki notation. However, for the *CFT-VM*, constraints should be defined apart in the *CFT-CM*.
3. **Variants' configuration stage** (Figure 9, mark C) Modelers should identify (1) which alternatives (from those specified in the *CFT-VM*) make up each variant and (2) what features of the *TRE-VM* are appropriate for this variant. This is specified by means of the definition of the *Resolution Model*. It contains the specific *replacements* for each *design placement* of the model, the appropriate features that configure the *TRE-VM* for the variant as well as the value of the context variables that define this process variant. This model contains only the resolutions that are applied at design time since their associated conditions depend on the initial context of the process. The resolutions that are applied at run time (i.e. those resolutions whose associated conditions depend on the current context of a process instance) will be handled during process execution, and a strategy to cope with them is necessary. However, this is out of the scope of this work since it only focus on BP variability at modeling time.

Following the stages of this process, modelers are able to properly put into practice the proposed approach. Thus, it is ensured the correctness of the resulting BP models regarding the approach. In addition, this process is completely independent if modelers start the modeling process from the scratch or from existing models that already include BP variations. This is due to the fact that modelers always need to identify process commonalities and individualities directly from the domain being modeled.

6 Related work

This section describes a short review of the contemporary BP variability modeling approaches. These approaches were developed due to the limitation of the original BPMLs (EPC or BPMN) to model properly BP variability. Afterwards, we present a short discussion about how they support BP variability modeling.

6.1 Contemporary approaches to model BP variability

C-EPC (Configurable EPC)[3] is an extension of EPC (Event-driven Process Chain) that, by means of new constructs (*configurable nodes* and *configuration requirements and guidelines*), represents variability in reference process models[2]. Within one model, *configurable nodes* (functions and connectors) allow specifying different choices depending on the context of use. The *configuration requirements and guidelines* allow stating, through logical predicates, the valid model configurations. The original language support was extended to consider other elements such as roles and objects [15]. In line with C-EPC, we find other proposals such as C-YAWL [16] (based on *hiding and blocking*) or Feature-EPC [17] (based on *feature models*). The goal of all of them is to perform a configuration of one reference process model to obtain an individualization of it for a specific context. It is also worth mentioning the Questionnaire approach [18] which, by means of questionnaires models, guides and prevents users from making inconsistent choices during the configuration task.

PESOA [4] is a project for the development and customization of families of process oriented software. As a result, taking into account the relevance of the reusability aspect, a set of annotations (stereotypes) is identified to capture the different semantics that variability may have within the model. Thus, models gather all variants' commonalities and individualities, which are identified by this set of annotations. The annotations have already been transferred to different languages such as UML Activity Diagrams or BPMN.

Provop [5] is an operational approach conceived to support BP variability during the process life cycle, which involves both design and run time. In Provop, model configuration is performed by adjusting a base model to a given context through a set of high-level change operations (INSERT, DELETE, MOVE and MODIFY). These change operation can be grouped into *options* to specify more complex adjustments. In turn, it is also possible to specify relationships between these options to define their semantics of use. The options that constitute a process variant are selected after evaluating a given context, which is represented explicitly in a model by means of *context variables*.

6.2 Discussion about BP variability modeling support

The described approaches have been evaluated against a set of desirable criteria [19]. These criteria are defined based on the objectives SPL is looking for (i.e. *commonality capitalization* and *variation management*). The criteria consider how BPMLs are endowed with new expressiveness to identify model parts related to variability. Also, they consider how BPMLs facilitate model management, especially regarding variability. To carry out this evaluation we have developed a set of real case studies applying the different approaches[3]. Table 1 shows a brief summary of it. The used symbols are: "+" (full support), "+/-" (partial support), "-" (no support) and "na" (not applicable).

[2] models that formalize recommended practices for a certain domain

[3] Technical Report PROS-TR-2011-03 `http://www.pros.upv.es/index.php/en/informes-tecnicos`

Desirable criteria	C-EPC	PESOA	Provop
C1 Variation points specification	+	+	+/-
C2 Alternatives for the variation points spec.	-	+	+
C3 Context of the alternatives spec.	-	+	+
C4 Relationships between alternatives spec.	-	+/-	+
C5 Language expressiveness regarding variability	+	-	-
C6 Process context regarding variability spec.	-	+	+
C7 Variation point resolution time (design/run time) spec.	na	-	+

Table 1. Summary of the approaches evaluation

From this table we can extract that from a modeling point of view, C-EPC, PESOA and Provop do not provide support for the desirable criteria. In C-EPC, the reason for not satisfying C2, C3, C4, and C6 is mainly due to the fact that variability concepts are not introduced into the language as first-class elements. For instance, the alternatives that fit in the variation points (represented by C2) are not explicitly defined in the model since they are integrated in one model. They have to be deduced by analyzing the different configurations defined for each configurable node (function or connector). As a consequence, modelers cannot explicitly define relationships between these alternatives either (C4). The same occurs with their context (C3) and process context (C6). Even though variant conditions are specified by means of *configuration requirements* and *configuration guidelines* associated to *configurable nodes*, these conditions are expressed in terms of the configuration of other model elements, not making explicit the underlying condition for a specific variant to be instantiated. In contrast to C-EPC, in PESOA and Provop we find mechanisms to specify variability as first-class elements of the language. However, they delimit their scope to addressing variability only regarding control flow and tasks, not considering variability associated to other elements, such as roles or objects (C5). Regarding the specification of variation point resolution time (C7), it does not apply to C-EPC since this approach is designed only for model configuration prior to model deployment. Regarding PESOA, it does not provide mechanisms to differentiate in the model the variation point resolution time, which would define the set of decisions that need to be taken prior model deployment.

Nevertheless, the approach we propose has been defined to achieve the SPL objectives, taking into account the defined desirable criteria. Thus, the approach allows specifying variation points by defining *placements* (C1), the alternatives that fit in all these points by defining *replacements* (C2), the set of variables that contextualize a specific variant by defining the *Resolution Model* (C3, C6), the relationships between these alternatives by defining constraints in *CFT-CM* and *TRE-VM* (C4), and the resolution time in which a variation point needs to be solved by specializing the *placement* concept into *design time* and *run time placements* (C7). In addition, by the refinement of the *Variation Model*, the approach considers not only the variability that can occur regarding control flow, but also the variability that appears regarding other modeling elements (C5). However, after developing real case studies with the proposed approach, we are

aware of the fact that a tool is needed to manage BP variability with this type of approach. Thus, we are currently developing a graphical editor that will help modelers to properly manage the complexity introduced by the defined models. By interrelating properly these models, the tool will reduce the fragmentation problem that may appear when handling them.

7 Conclusions and further work

In this work we have proposed an approach to deal with BP variability modeling, that while applied to BPMN, can be transposed to other notations. The cornerstone of the approach is the separation of variants' commonalities from individualities. This separation allows modelers not only avoiding duplication and divergence of variants' commonalities, but also making explicit location, rationale and dependences between variations. To cope with such separation, we have based on the BVR approach. This approach proposes to deal with variability by modeling separately variants' commonalities, individualities, and configurations. Specifically, to cope with the variability that arises according to different modeling elements, we have refined the model that refer to model individualities. This refinement solves the scalability problem that appears when the BP been modeled involves variability in different modeling elements. In addition, a methodology to put into practice the approach has been defined. This methodology has been used to develop a case study, which illustrates the applicability of the approach. Finally, a case study evaluation has been perform, showing the strengths and weaknesses of the approaches.

This work is been developed as part of a bigger project aimed at supporting BP variability during the entire BPM lifecycle. Therefore, the next step is to define model transformations to convert the defined models into a process-engine deployable representation. This implies also defining a strategy to cope with the variability that needs to be solved at run time. For such purpose, we will consider other approaches (such as [20]) that already deal with variability at run time. In particular, we are considering to explore the possibility of adopting our experience in systems' reconfigurations based on models at run time ([21]).

References

1. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Number 978-3-540-73521-2. Springer-Verlag Berlin Heidelberg 2007 (2007)
2. Mendling, J., Reijers, H.A., van der Aalst, W.M.P.: Seven process modeling guidelines (7pmg). Information & Software Technology **52**(2) (2010) 127–136
3. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. Inf. Syst. **32**(1) (2007) 1–23
4. Puhlmann, F., Schnieders, A., Weiland, J., Weske, M.: Variability mechanisms for process models. Technical report, BMBF-Project (2006)
5. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: The provop approach. Soft. Process: Improvement and Practice (2010)
6. Clements, P.: Software product lines: practices and patterns. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2001)

7. Cetina, C., Giner, P., Fons, J., Pelechano, V.: Autonomic computing through reuse of variability models at runtime:case of smart homes. Comp. **42**(10) (2009) 37–43
8. Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Peterson, A.S.: Feature-oriented domain analysis (foda) feasibility study. Technical report, Carnegie-Mellon University Software Engineering Institute (November 1990)
9. Bayer, J., Gerard, S., Haugen, Ø., Mansell, J.X., Møller-Pedersen, B., Oldevik, J., Tessier, P., Thibault, J.P., Widen, T.: Consolidated product line variability modeling. In: Software Product Lines. (2006) 195–241
10. Haugen, O., Moller-Pedersen, B., Oldevik, J., Olsen, G., Svendsen, A.: Adding standardized variability to domain specific languages. In: Software Product Line Conference, 2008. SPLC '08. 12th International. (2008) 139 –148
11. CVL-RFP: `http://www.omg.org/cgi-bin/doc?ad/2009-12-3` (2009)
12. Hallerbach, A., Bauer, T., Reichert, M.: International Handbook on BPM. In: Configuration and Management of Process Variants. Springer (2010)
13. Czarnecki, K., Helsen, S., Eisenecker, U.W.: Formalizing cardinality-based feature models and their specialization. Soft. Proc.: Improvement and Practice (2005) 7–29
14. Jablonski, S., Bussler, C.: Workflow Management: Modeling Concepts, Architecture and Implementation. Intern. Thomson Computer Press (September 1996)
15. Rosa, M.L., Dumas, M., Hofstede, A., Mendling, J., Gottschalk, F.: Beyond control-flow: Extending BP configuration to roles and objects. In: ER. (2008) 199–215
16. van der Aalst, W.M.P., Dreiling, A., Gottschalk, F., Rosemann, M., Jansen-Vullers, M.H.: Configurable process models as a basis for reference modeling. In: Business Process Management Workshops. (2005) 512–518
17. Vervuurt, M.: Modeling business process variability : a search for innovative solutions to business process variability modeling problems. Student Theses of University of Twente. October 2007 (2007)
18. Rosa, M.L.: Managing Variability in Process-Aware Information Systems. PhD thesis, Faculty of Science and Technology Queensland University of Technology Brisbane, Australia (2009)
19. Ayora, C., Torres, V., Pelechano, V.: Dealing with Variability in Business Process Models: An Evaluation Framework. Technical report, ProS-TR-2011-11, ProS-UPV (2011)
20. Adams, M.J.: Facilitating Dynamic Flexibility and Exception Handling for Workflows. PhD thesis, Fac. of Inf. Tech. Queensland University of Technology (2007)
21. Alferez, G., Pelechano, V.: Context-aware autonomous web services in software product lines. In: Proceedings of the 15th International Software Product Line Conference. SPLC '11 (2011) 100–109