UNIVERSIDAD POLITECNICA DE VALENCIA

PROS
Centro de Investigación en Métodos de Producción de Software

Informe Técnico / Technical Report




Dealing with Variability in Business Process Models: An Evaluation Framework

Clara Ayora, Victoria Torres, Vicente Pelechano





Ref. #:	ProS-TR-2011-05				
Title:	Dealing with Variability in Business Process Models: An Evaluation Framework				
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:	16
Release date:	February 2011				
Key words:	Business Process Modelling, Variability Modelling				

Dealing with Variability in Business Process Models: An Evaluation Framework*

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 are representations of real-world BP that gather relevant aspects for a specific purpose, usually organization or information system design. Apart from very specialized and concrete BPs, there is a big number of BPs, e.g. buying and selling, that are shared at a high level of abstraction by different domains but differ in the way BPs are realized, e.g. differences that can arise due to disparate reason such as legal regulations, culture traditions, organization resources, etc. Related with the commonalities and differences that can appear during the BP modelling process, different approaches have been developed from the BP Modelling (BPM) research community. This paper gathers the most extended and well-known approaches and presents an evaluation framework to compare and evaluate them for their suitability for variability modelling.

Keywords: Business Process Modelling, Variability Modelling

1 Introduction

Business Process models represent, by means of tasks and resources, how organizational goals are achieved. These models are commonly built either for organization design purposes 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.

However, BP modelling is not a trivial task [1].It requires, not only mastering the problem domain being represented, but also the language or notation that is being used to perform such representation. In addition to this complexity, the different variations of a specific model based on the application context[1] turn models into artifacts that are more complex to build, manage and to understand.

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

[1] e.g. when models are built for the international market and require their adaptation for different legal or cultural environments

In these cases, considerations such as process granularity, context dependency or scalability turn into first order requirements during the modelling process.

In order to deal with this complexity and to facilitate the construction and management of such models, many efforts from the BPM research community have been conducted ([2], [3], [4], [5], [6], [7]). In these works, by the use of different techniques (e.g. feature models, ripple-down rules) and approaches (e.g. operational, model projection) authors deal with the problems that arise during the variability modelling process. In addition, some of these works also deal with the management of variability during run time, which involves considering also process evolution techniques such as the ones presented by *Weber et al.* in [8].

In this work, we have considered the approaches that have been designed to deal with business process variability modelling, independently of the mechanisms used for such purpose. The objective is to provide a clear picture of the different possibilities while variability modelling and also understanding the suitability of each approach according to the context of use. For such purpose, we have selected representative approaches found in literature and have analyzed them based on a set of evaluation criteria, which reflect the issues that need to be addressed in order to handle properly BP variability at the analysis/design phase. This evaluation criteria has been defined based on a refinement of the requirements already stated in [9] to support process variants, and in [10] to deal with the modelling language quality. These criteria are presented as an evaluation framework which allows us to identify the strengths and the weaknesses of the selected approaches. In addition, the framework can help organizations in the decision of which approach suits better to their needs (the BP modelling process of each organization may vary according to its characteristics, e.g. starting the modelling process from scratch, level of expertise of its business modellers, etc.). The framework can be applied to any proposal since it is not focused on BPML concrete aspects but focused on the variability concepts themselves.

The remainder of the paper is structured as follows. Section 2 describes the impact that variability has within the analysis/design phase of the BP lifecycle. Then, section 3 presents the evaluation criteria that make up the framework and that it is used to evaluate the different approaches. Section 4 proceeds with such evaluation and provides for each of the evaluated approaches a brief description about its main characteristics. Finally, based on the approaches previous analysis, section 5 provides an analysis of the evaluation and presents the conclusions that can be extracted from it.

2 Variability within the BPM analysis/design phase

Within the four main phases that make up the BPM lifecycle [11], in this work we focus on the first one which deals with BP analysis/design. Specifically, we reflect on the impact of dealing with BP variability on this phase. During the analysis/design phase, and based on domain requirements, BPs are identified, review, validated and represented by means of BP models. These models are built by means of Business Process Modelling Languages (BPML) such as Petri nets,

EPC, YAWL, BPMN or UML Activity Diagrams. However, the original constructs provided by these languages do not provide enough expressivity to deal properly with BP variability (e.g. the Boolean conditions do not allow describing the nature of the different choices [12]). Therefore, BPMLs should allow representing model variability as such, bringing the concepts related to variability as first order concepts of the language.

In addition, another important aspect not directly related to the BPML is the methodology provided to build such models. Proposals should not only focus on providing expressivity to deal with the definition of variability in BPs but also to guide during the construction of the corresponding models [13]. Moreover, taking into account the heterogeneity of the situations in which BP models are built, this guidance should be flexible enough to accommodate not only the different types of stakeholders (which may have different background and level of expertise) but also the context in which the model is built (e.g. the procedure is different depending if we start from scratch or from an already defined model).

When dealing with BP variability, once BP models are built, these need to be individualized to be used within a specific context. To achieve this individualization, we first have to configure such models by stating which choices are going to be included in the individualized model and which choices are going to be discarded. These decisions can be taken either at design time or/and at run time. In some cases, BP models formalize recommended practices for a certain domain (such as reference process models) and need to be individualized prior their use. In this case, model configuration is performed at design time. However, in other cases, the configuration depends on the process current context (e.g. in a hospital, the urgency procedure depends on the characteristics of the patient being assisted) and decisions have to be postponed at run time. In both cases, it is necessary to define a set of rules (normally expressed in a formal language) that establish which choices have to be selected for a specific context.

Therefore, in order to face properly the task of BP variability modelling, BPMLs should provide enough expressivity to deal with all of these modelling and individualization aspects. In order to cover this expressivity, the following questions need to be considered:

- Which are the points (*variation point*) within the BP model that accept several alternatives (*variants*)?
- During which phase within the BP lifecycle these points should be solved?
- Which variants fit into each variation point?
- Which process variables determine the variants selection?
- Which semantics apply for a set of variants for a specific context (e.g. two or more variants should be used always together)?
- Which steps should be followed to define process model commonalities and process model individualities?
- Which steps should be followed to configure and individualize process models?

Based on these questions, in the following section we present an evaluation framework to analyze the support provided by the most-well known approaches to deal with BP variability modelling issues.

3 Evaluation framework

To evaluate the different approaches selected in this paper, we first need an evaluation framework that establishes the concepts that BPMLs need to consider to deal properly with variability during the BPM analysis/design phase. The lack of such framework has motivated us to propose an evaluation framework that specifies the set of properties that should be satisfied by any approach to deal properly with BP variability. These properties have been specified and defined by answering the questions presented in the previous section and using the requirements already stated by Hallerbach et al. in [9] and by Nysetvold et al. in [10]. The evaluation criteria that we propose have been organized in two building blocks which refer to: (1) concepts and (2) quality factors. Meanwhile the first block gathers a set of concepts that need to be addressed by BPMLs to deal properly the BP modelling task, the second block gathers desirable properties that contribute positively to improve the approach/BPML regarding its adoption.

- **Concepts**. In order to bring variability concepts as first class constructs, BPMLs should provide the mechanisms that allow the specification of such concepts. These concepts constitute the first block of the evaluation criteria and their definition is presented below:
  - **EC1. Variation point**
    A variation point is defined as a precise position within a process model that admits different possibilities according to the current context or situation.
  - **EC2. Variant**
    A variant is defined as an alternative process fragment that can be applied in a specific *variation point* for a particular context.
  - **EC3. Variant context**
    A variant context is defined by the environment conditions that make a particular variant to be instantiated.
  - **EC4.Variant relationships**
    A variant relationship is defined as a constraint between two or more *variants* that establishes the proper use of the involved *variants* within a specific context (e.g. mutual exclusion or implication).
  - **EC5. Language variant scope**
    The support provided by the language to specify variability should include any of the elements that make up a process model (control flow, physical objects, data, roles, events, etc.).
  - **EC6. Process context regarding variability**
    The process context is defined as the set of variables that determine the current state of a BP, in particular, according to BP variability.

- **EC7. Variation point resolution time**
  The language should allow modellers to distinguish between *variation points* whose resolution depends on the initial context (design time) or on the current context of an instance process (run time).

– **Approach quality factors**. This block gathers a set of quality factors that are desirable in the different BPMLs to ensure their successful adoption.

  - **EC8. Flexibility**
    In order to deal with *variants* evolution, the language should provide mechanisms to easily make changes over the model, in particular those parts related to *variants*.
  - **EC9. Scalability**
    The language should provide techniques that allow evolving BP models without losing the ability to handle a great number of model variants [2].
  - **EC10. Legibility**
    The language should provide graphical elements to easily distinguish between model commonalities and model individualities.
  - **EC11. Understanding**
    The language should provide abstractions close to the concepts used to represent BP variability. A correct abstraction of these concepts will result in more easily understandable models.
  - **EC12. Low learning curve**
    The language should be easy to learn. It is easier to learn a language when it is based on a well-known and commonly used language. Moreover, if this well-known language is a standard language, we can take advantage of its benefits such as existing tools, guidelines, etc.
  - **EC13. Separation of Concerns**
    The language should allow modellers to manage separately *variants* from the base model (model that gather commonalities). This separation contributes in reducing the change impact on the process model, reusing the existing *variants*, and improving the understanding of the process model ([14],[15]).
  - **EC14. Tool support**
    The language should be supported by a tool since this support is usually critical for the language adoption. In addition, it facilitates the management of the variability, specifically during the analysis/design phase.
  - **EC15. Guideline support**
    The existence of guidelines that assist users during variability management in BP models facilitates the learning process and the use of the language. Specifically, this guidelines are desirable during the analysis/design phase while defining model commonalities, variants, and configuration. As a consequence, it turns the modelling process into a more objective task, independent of the modeller skills or experience.

[2] a model variant represents an individualization of the process model

4 Proposals evaluation

In the previous section we have defined an evaluation framework with the main properties that should be satisfied to deal properly with BP variability. The goal of this current section is to provide a brief explanation of the most relevant approaches that deal with BP variability and to evaluate them against the framework presented before.

The approaches that have been evaluated in this paper are C-EPC [2], PESOA [16], Provop [17], and Worklets [7]. All these approaches have been put into practice (see *Case Studies Development*[3]) by means of three different case studies from different domains, all extracted from the reviewed literature. These case studies include (1) a service process handling vehicle repair in a garage [18], (2) a simplified version of a healthcare process representing a cruciate rupture treatment [8], and (3) an e-business shop [16]. This exercise has allowed us to better understand BP variability requirements at the modelling level and also to detect some deficiencies of the analyzed approaches while applying the different case studies.

C-EPC (Configurable EPC) C-EPC is an extension of EPC (Event-driven Process Chain) that includes new constructs to represent variability in reference process models [2]. The main idea of C-EPC is to represent differently model commonalities from individualities in order to configure the process model according to its context. This differentiation is possible by combining the use of (1) *configurable nodes* (functions and connectors), which allow specifying different behaviour depending on the context of use with (2) *configuration requirements and guidelines*, which state, by means of logical predicates, the valid configurations of the model.

In order to deal with variability in BP models following the idea of C-EPC, other approaches, such as C-YAWL [12] (based on (*Hiding and Blocking*) or Feature-EPC [5] (based on *Feature Models*) have been proposed. The goal of all of them is to perform a configuration of the process model to obtain an individualization of it within a specific context. Since C-EPC is widely known and extended, we have decided to evaluate it as a representation of this group of approaches.

The evaluation of the C-EPC approach against the evaluation criteria defined in section 3 is presented below:

- EC1. *Variation Point:* It is possible to clearly identify them by means of *configurable functions* and *connectors*.
- EC2. *Variant:* It is not possible to clearly identify from the model which variants are being considered in the process. This information is scattered among model functions, events and connectors.

[3] `http://arxiv.org/vc/arxiv/papers/1102/1102.4518v1.pdf`

- EC3. *Variant Context:* It is possible to express it by means of configuration requirements and guidelines which are expressed as logical predicates.
- EC4. *Variant relationships:* It is not possible to define explicitly relationships between variants since C-EPC does not allow defining variants as such. Moreover, although configuration requirements define relationships between functions and connectors, this information is scattered among the entire model which implies that these relationships are not neither clear nor transparent to the modellers.
- EC5. *Language variant scope:* It is possible to consider variability aspects in other model elements. The provided expressivity to support variability by the original approach (which includes events, functions, logical operators and dynamic connectors) is extended in [19] to consider also roles and objects.
- EC6. *Process context regarding variability:* It is not possible to define the process context regarding variability. C-EPC does not provide enough expressiveness to explicitly define the set of variables that determine the process context. Again it is scattered among *configuration requirements*.
- EC7. *Variation point resolution time:* It is possible to distinguish when variation points are solved due to their resolution is done at design time before process deployment. This approach does not take into account variability that may occur at runtime.
- EC8. *Flexibility:* It is not possible to evolve or change C-EPC models easily since they are integrated representations (of model commonalities and individualities) and variants are not clearly identified, e.g. one small change in a specific part of the model may entail big changes along the whole model.
- EC9. *Scalability:* It is not possible to manage a great number of variants. Since C-EPC does not provide any techniques to support model variant evolution, the bigger number of them, the more difficult they become to handle.
- EC10. *Legibility:* It is partially possible to distinguish model commonalities and individualities since model variants are not easily extracted from the diagram. However, model elements related to model variability (i.e. configurable nodes and configuration requirements and guidelines) can be identified graphically in the model.
- EC11. *Understanding:* It is possible to understand C-EPC since it provides the appropriate abstractions to represent model variations. These abstractions are configurable nodes (which can be valued as included, excluded or conditionally skipped) and configuration requirements and guidelines.
- EC12. *Low learning curve:* It is partially possible to learn the language quickly. The learning process of C-EPC is fast due to the proximity to the already existing EPC elements (the introduced configurable nodes are small variations to the original nodes). However, this high speed is not associated to the use of requirements and guidelines, which forces the modeller to know logical predicates.
- EC13. *Separation of Concerns:* It is not possible to manage separately variants from the base model. The C-EPC approach forces to handle jointly model commonalities and individualities.

– EC14. *Tool support:* It is not possible find a tool to design C-EPC models. A designer for C-EPC models is not available yet. Nevertheless, in [20] is presented a tool (C-EPC Validator) to check the compliance of a C-EPC model configuration against configuration requirements and guidelines.
– EC15. *Guideline support:* It is not possible to find clear guidelines (that assists modellers during model design and configuration) despite the fact that there exist documentation about the language ([2]). In fact, to guide users and to reduce the complexity of C-EPC models configuration task, La Rosa defined a questionnaire-driven approach in [4].

Variant-Rich Process Models (within the PESOA project) PESOA (www.pesoa.org) is a cooperative project carried out by a group of companies and academics whose main objective was the investigation of an approach for the development and customization of families of process oriented software. As a result, focused on the design level and taking into account the relevance of the reusability aspect, a set of *basic* and *composite* variability mechanisms were identified [16]. The *basic* set includes (1) encapsulation of varying subprocesses, (2) addition, replacement, omission of encapsulated subprocesses, (3) parameterization, and (4) variability in data types, while the *composite* includes (5) inheritance, (6) design patterns, and (7) extensions/extension points. This set was transferred to different languages such as UML Activity Diagrams, UML State Machines, BPMN, and Matlab/Simulink. Since we are focused on the concepts and not in the particularities of any language, in this survey we have taken one of them, in particular BPMN.

The evaluation of the PESOA approach against the evaluation criteria defined in section 3 is presented below:

– EC1. *Variation Point:* It is possible to identify them in the model by means of the introduction of the «VarPoint» label (stereotype) which is attached to the places (i.e. tasks) where variability may occur. According to their expected behaviour, these labels may vary into one of the following stereotypes: «Abstract» and «Null».
– EC2. *Variant:* It is possible to identify the variants in the model by the introduction of the «Variant» stereotype which is attached to those tasks that can fit in a variation point. This stereotype may change into the stereotypes «Default», «Alternative» and «Optional» depending on the variant behaviour.
– EC3. *Variant Context:* It is possible to express it within the model. Variants are refined with two tagged values, one of which represents the specific feature the variant belongs to. In addition, the variant context can be specified by means of the parameterization of the different attributes (values that are used in the model to activate different flows, e.g. a condition expression).
– EC4. *Variant relationships:* It is partially possible to specify relationships between variants due to the little support provided. Only two different types of relationships are defined: *implementation*, to associate possible resolutions

to the variation points, and *inheritance*, to express alternative behaviour which adds or removes elements regarding specific rules.

- EC5. *Language variant scope:* It is not possible to support variability issuess in other model elements. PESOA just considers the variability that can arise associated to control flow and task elements, leaving out other elements such as roles or events.
- EC6. *Process context regarding variability:* It is possible to express the process context by means of the variant feature values associated to the tasks. These features define explicitly the state of the process.
- EC7. *Variation point resolution time:* It is not possible to distinguish if a variation point is solved at design time (depending on the initial context) or at run time (depending on the current instance context).
- EC8. *Flexibility:* It is not possible to evolve or change variants easily. Since process model commonalities and individualities are defined together, changes in variants may imply reconsidering other parts of the model, which difficults variant evolution.
- EC9. *Scalability:* It is not possible to handle a great number of variants since process model commonalities are modelled jointly with model individualities. The higher level of variants a model has, the more difficult they become to handle.
- EC10. *Legibility:* It is possible to differenciate graphically model commonalities from individualities. Despite variants are integrated in the model, the enrichment of task/subprocess elements by means of stereotypes allows this differentiation.
- EC11. *Understanding:* It is not possible to comprehend easily the language. The stereotypes introduced to represent concepts related to BP variability does not resul suitable. Their associated semantics is broader than the semantics of the word which represents the stereotype. For instance, the «Null» stereotype indicates not only task/subprocess optional behaviour (behaviour also associated to the «Optional» stereotype) but also restricts to one the number of possible variants associated to the variation point.
- EC12. *Low learning curve:* It is not possible to learn the language easily. The broad semantics associated to the introduced stereotypes takes time to learn and use them properly, which results in a steep learning curve.
- EC13. *Separation of Concerns:* It is partially possible to separate model commonalities from individualities. The PESOA approach forces to handle them jointly but variation points and their different alternatives are labelled through stereotypes which allow distinguishing and managing individualities separately from commonalities.
- EC14. *Tool support:* It is possible to find a tool that supports the language. The approach has been implemented as a plug-in of the Eclipse IDE. This tool provides support for configuring a feature diagram and for applying the results to the process model.
- EC15. *Guideline support:* It is not possible to find explicit guidelines that assist during the modelling process despite the fact that there exist documentation that explains the approach ([16], [3]).

Provop (PROcess Variants by OPtions) Provop is an operational approach for managing large collections of process variants during the process life cycle. It has been motivated by the fact that a process variant can be created by adjusting (configuring) a given process model to a given context. These variant-specific adjustments are expressed by means of a set of high-level change operations (INSERT, DELETE, MOVE and MODIFY) [17]. Furthermore, Provop allows more complex process adjustments by grouping multiple change operations into so called *options* [18]. Thus, a particular process variant is specified (configured) by applying one or more options to the respective base process. The options used for a process variant are selected when evaluating the given context. Provop provides a model for capturing this process context by means of *context variables*, which represent different domain dimensions of it.

The evaluation of the Provop approach against the evaluation criteria defined in section 3 is presented below:

- EC1. *Variation Point:* It is partially possible to identify them within the model. When it refers to INSERT and DELETE model changes, variation points can be clearly identify by means of the *adjustment points* which are represented graphically in the diagram by black diamonds. However, this does not occur when the operation refers to MOVE or MODIFY changes.
- EC2. *Variant:* It is possible to identify them clearly by means of the specification of the proposed change operations.
- EC3. *Variant context:* It is possible to represent it by means of the context variables which are defined by a name, a description, a value range and a mode (*static* or *dynamic* depending on whether its value changes or not during the execution of the process).
- EC4. *Variant relationships:* It is possible to define relationships such as implication, mutual exclusion, application order, hierarchy and at-most-n-out-of-m-options [21] between the options that that make up a model variant.
- EC5. *Language variant scope:* It is not possible to manage variability aspects in other model elements. Despite the fact that it is based on EPC, Provop only provides variability support for the basic elements of it (functions, logical operators and dynamic connectors) and not for other elements such roles and events.
- EC6. *Process context regarding variability:* It is possible to express it by means of the context variables, which capture the different domain dimensions of the context.
- EC7. *Variation point resolution time:* It is not possible to distinguish when variation points are solved. Even though Provop provides context rules for determining the variants that can be applied for a specific variation point, it cannot be explicitly defined whether variation points depend on the initial context or on the current process instance context.
- EC8. *Flexibility:* It is possible to evolve or change models easily by means of the definition of new options, e.g. to obtain a new variant of the process it is only necessary to define a new option that changes the model in a different way from the already existing ones.

- EC9. *Scalability:* It is possible to evolve models that include a great number of variants due to change operations and options, which facilitate model change.
- EC10. *Legibility:* It is possible to identify clearly model commonalities and individualities since they are specified separately. Model commonalities are represented in a base model which may be modified by means of operations to obtain a process model individualization.
- EC11. *Understanding:* It is possible to understand the language since Provop provides an adequate set of abstractions to represent model variability. These abstractions are adjustment points (to identify variation points within the model), change operations (to define the difference between the basic process model and its individualization), and options (to group change operations) which result suitable to represent such concepts.
- EC12. *Low learning curve:* It is possible to learn Provop easily. The learning curve of Provop is low due to the simplicity of the change operations (INSERT, DELETE, MODIFY and MOVE) and the way model variants are built.
- EC13. *Separation of Concerns:* It is possible to manage separately variants from the base model. The Provop approach provides a proper separation of concerns since variants are clearly defined and identified by means of the change operations.
- EC14. *Tool support:* It is possible to find a proof-of-concept prototype. It enhance an existing BP modelling tool (ARIS Business Architect) with the facilities for process variant configuration and management.
- EC15. *Guideline support:* It is possible to find guidelines that explain step by step how to model and configure BP models [18]. Besides, although the main document with the full explanation of the language is not written in English [6], there exist documentation ([22], [17]), that gives a complete and formal description of the approach.

Worklets This proposal is an approach for dynamic flexibility, evolution and exception handling in workflows through the support of flexible work practices. It was not conceived to be targeted to any notation which means that it can be applied to any BPML. A *worklet* is defined as a small, complete and re-usable workflow specification which handles one specific task in a composite parent process [7]. In this parent process, an *extensible repertoire* (or catalogue) of worklets is maintained for each nominated task. Each time a worklet is needed, an intelligent choice is made from its repertoire using a set of associated *selection rules* (Ripple Down Rules, RDR). These rules determine the most appropriate substitution [23]. Then, the selected worklet is launched as a separate case and, when it has completed, the control is returned to the original (parent) process, which continues normally. Thus, dynamic ad-hoc change and process evolution are provided without having to modify the original process specification and/or to resort to off-system intervention [24].

The evaluation of the Worklet approach against the evaluation criteria defined in section 3 is presented below:

- EC1. *Variation Point:* It is not possible to identify them clearly in the model. The nominated tasks (named worklet-enabled tasks), that may be replaced by worklets, do not have any marks to be distinguished from the non-worklet-enabled tasks.
- EC2. *Variant:* It is possible to identify them easily since a worklet is a process fragment that can fit in a variation point. Thus, each worklet as such constitutes a variant.
- EC3. *Variant Context:* It is possible to specify the specific context that make a worklet to be instantiated by means of the conditions of its related RDR.
- EC4. *Variant relationships:* It is partially possible to define variant relationship. Only one relationship between worklets is present within the approach: mutual exclusion relationship, which is implicitly defined through the RDR, i.e. each "true" branch excludes its related "false" branch in the tree.
- EC5. *Language variant scope:* It is not possible to cover variability aspects in other model elements since the worklet concept only refers to tasks or, more generally, to process fragments. Other elements such as roles or events are not considered for variability issues.
- EC6. *Process context:* It is possible to specify the process context by means of the conditions of the RDR of the worklets.
- EC7. *Variation point resolution time:* It is possible to distinguish when variation points are solved. Their resolution is done always at run time through the evaluation of the RDR. This approach does not take into account variability at design time.
- EC8. *Flexibility:* It is possible to change the models easily since new worklets can be added to the repertoires. Thus, the process model undergoes a dynamic natural evolution.
- EC9. *Scalability:* It is possible to handle a great number or variants since variants are defined separately from the base model and gathered in repertoires.
- EC10. *Legibility:* It is not possible to distinguish process model commonalities from model variations since variation points are not clearly identified in the diagram.
- EC11. *Understanding:* It is possible to understand the RDR due to its tree conceptual structure.
- EC12. *Low learning curve:* It is possible to learn the language easily. The novelty of the Worklet approach relies on the use of RDR to perform the selection of the appropriate choice. The definition of such rules is quite simple and intuitive, whose learning process does not imply a big effort.
- EC13. *Separation of Concerns:* It is possible to manage separately variants from the base model since worklets are defined in repertoires separately from the base process.
- EC14. *Tool support:* It is possible to find a tool that supports the approach. The Worklet service, named *Worklet Dynamic Process Selection Service*,

has been developed as a YAWL Custom Service. It was implemented to ensure proof-of-concept and also to construct the framework for the overall approach.
– EC15. *Guideline support:* It is possible to find enough documentation to support the modelling and the configuration process tasks ([7], [24], [23]) despite the fact that an explicit manual of the modelling phase does not exist.

5 Analysis of the evaluation results

In the previous section, we have evaluated the most relevant approaches that deal with variability at the analysis/design phase. The results of this evaluation is summarized in Table 1, where a set of symbols has been used to specify the fulfilment of each evaluation criterion: a "+" indicates that the approach fulfils completely the evaluation criterion, a "-" indicates that it is not fulfilled, and a "+/-" indicates that it is partially fulfilled.

	EC1	EC2	EC3	EC4	EC5	EC6	EC7	EC8	EC9	EC10	EC11	EC12	EC13	EC14	EC15
C-EPC	+	-	+	-	+	-	+	-	-	+/-	+	+/-	-	-	-
PESOA	+	+	+	+/-	-	+	-	-	-	+	-	-	+/-	+	-
Provop	+/-	+	+	+	-	+	-	+	+	+	+	+	+	+	+
Worklets	-	+	+	+/-	-	+	+	+	+	-	+	+	+	+	+

Table 1. Summary of the approaches evaluation.

5.1 Concepts analysis

According to the ECs related to the first block (EC1-EC7), only the environment conditions that make a variant to be instantiated (EC3) is satisfied by all the approaches. This makes sense since this information is absolutely necessary to obtain an individualization of the process model. Nevertheless, this unanimity is not maintained for the remaining ECs since they are not considered as first class elements of the BPML. As Table 1 shows, depending on the approach and on the concrete aspect, ECs are supported or not.

5.2 Quality factors analysis

Regarding the second block of ECs (EC8-EC15), their fulfillment implies that the approach is more likely to be adopted by users and industry to deal with BP variability at analysis/design time. In this case, none of the EC is satisfied by the considered approaches. For instance, flexibility (EC8) and scalability (EC9)

are not prevalent aspects for C-EPC and PESOA, where model commonalities and individualities are modelled jointly. Providing an appropriate support for model variants evolution as well as being able to handle a big amount of them are desirable approach features to facilitate the maintenance of BP models along time. The lack of guideline support (EC15) that the approaches provide is also worth mentioning. The existence of guidelines contributes to improve the modelling process, turning it into a more objective task independent of the modeller skills or experience. Not providing such guidelines (as with C-EPC or PESOA) favours the construction of error-prone models not ensuring the quality of the resulting model.

5.3 Evaluated approaches analysis

A complete fulfillment of these evaluation criteria would mean that the approach is able to provide BP variability modelling coverage in a wide sense. This coverage implies considering variability modelling issues related for instance to model structure and behaviour, language elements (not limiting just to control flow elements) or model resolution time. The C-EPC approach, which deals with BP variability according to reference process models [4], is focused on obtaining process models individualizations before model deployment. For this reason, EC2, EC4, EC8, and EC9 are not satisfied by the approach. In addition, EC7 is not either satisfied since C-EPC does not consider variability issues at run time, which is important in domains where the process may change constantly. C-EPC is well recommended when starting from a reference process model where the objective is its adaptation for a specific domain (only the initial context of the process is considered). The PESOA approach was conceived in the context of the Software Product Lines. This explains that the approach satisfies almost completely the ECs related to variability concepts (EC1-EC6). On the other hand, the semantics associated to the stereotypes introduced by the approach does not result as intuitive as it may seem initially. In addition, the lack of modelling guidance does not contribute to clarify these semantics. In the Provop approach change operations and separation of concerns are key points to provide enough expressivity to handle with variability. However, Provop does not allow specifying when model variation points are solved. It neither provide support to specify variability associated to other BP elements such as roles or objects. These other elements are also very important within the BP and are subject to changes (e.g. in hospitals where in principle only doctors can carry out scans, nurses may be allowed to do them too). The Worklets approach was conceived to deal with process variability that appears due to process exceptions. Thus, it is focused on the variability that occurs at run time and not on the variability that is known at design time, which is also important when configuring the process model.

[4] models conceived to gather best practices for a specific purpose but considering many application contexts

References

1. Mendling, J., Reijers, H.A., van der Aalst, W.M.P.: Seven process modeling guidelines (7pmg). Information & Software Technology **52**(2) (2010) 127–136
2. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. Information Systems **32**(1) (2007) 1–23
3. Schnieders, A., Puhlmann, F.: Variability mechanisms in e-business process families. In: BIS. (2006) 583–601
4. Rosa, M.L.: Managing Variability in Process-Aware Information Systems. PhD thesis, Queensland University of Technology (2009)
5. Vervuurt, M.: Modeling business process variability : a search for innovative solutions to business process variability modeling problems (October 2007)
6. Hallerbach, A.: Management von Prozessvarianten. PhD thesis (2009)
7. Adams, M.J.: Facilitating Dynamic Flexibility and Exception Handling for Workflows. PhD thesis, Faculty of Information Technology. Queensland University of Technology (2007)
8. Weber, B., Sadiq, S.W., Reichert, M.: Beyond rigidity - dynamic process lifecycle support. Computer Science - R&D **23**(2) (2009) 47–65
9. Hallerbach, A., Bauer, T., Reichert, M.: Managing process variants in the process lifecycle. In: 10th Int'l Conf. on Enterprise Information Systems (ICEIS'08). (June 2008) 154–161
10. Nysetvold, A.G., Krogstie, J.: Assessing business processing modeling languages using a generic quality framework. In: Information Modeling Methods and Methodologies. (2005) 63–79
11. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Number 978-3-540-73521-2. Springer-Verlag Berlin Heidelberg 2007 (2007)
12. 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
13. Rosemann, M.: Potential pitfalls of process modeling: part a. Business Process Management Journal **12**(2) (2006) 249–254
14. Myllymäki, T.: Variability management in software product lines (12 2001)
15. Stan Bühne, Kim Lauenroth, K.P.: Modelling requi variability across product lines (November 2005)
16. Puhlmann, F., Schnieders, A., Weiland, J., Weske, M.: Variability mechanisms for process models. Technical report, BMBF-Project (2006)
17. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: The provop approach. Software Process: Improvement and Practice (2010)
18. Alena Hallerbach, Thomas Bauer, M.R.: International Handbook on Business Process Management. In: Configuration and Management of Process Variants. Springer (2010)
19. Rosa, M.L., Dumas, M., ter Hofstede, A.H.M., Mendling, J., Gottschalk, F.: Beyond control-flow: Extending business process configuration to roles and objects. In: ER. (2008) 199–215
20. C-EPC Validator: `http://www.mendling.com/EPML/C-EPC-Validator.xsl`
21. Hallerbach, A., Bauer, T., Reichert, M.: Managing process variants in the process lifecycle. In: 10th Int'l Conf. on Enterprise Information Systems (ICEIS'08). (June 2008) 154–161
22. Hallerbach, A., Bauer, T., Reichert, M.: Context-based configuration of process variants. In: 3rd International Workshop on Technologies for Context-Aware Business Process Management (TCoB 2008). (June 2008) 31–40

23. Adams, M., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Facilitating flexibility and dynamic exception handling in workflows through worklets. In: CAiSE Short Paper Proceedings. (2005)
24. Adams, M., Hofstede, T.A.H.M., Edmond, D., van der Aalst, W.M.P.: Implementing dynamic flexibility in workflows using worklets. Technical report, BPM Center Report BPM-06-06, BPMcenter.org (2006)