An Indexing Structure for Maintaining Configurable Process Models

Wassim Derguech, Gabriela Vulcu, and Sami Bhiri

DERI, Digital Enterprise Research Institute,
National University of Ireland, Galway
firstname.lastname@deri.org
http://www.deri.ie

Abstract. During the business process modeling phase, different environments and languages have been proposed. All of them are trying to narrow the communication gap between both business and IT users or making modeling task as optimal as possible. From these perspectives, we prioritize assisting business users to express in an efficient and easy way their requirements (i.e., defining their business process models). In this context, reusing existing process models is well supported and preferred rather than modeling from scratch. Configurable business process models aim at merging different process variants into a single configurable model. In this paper, we define an indexing structure to represent configurable process models. We give a set of structuring principles and we show how to maintain this structure when adding a new variant. We adopt a hierarchical representation of business goals variations. The contribution of our structure is that it allows for modularity handling.

Keywords: Business process, configuration-based modeling, goal hierarchy.

1 Introduction

Process Aware Information Systems (PAISs) [2] are used to manage and execute operational processes involving people, applications and data sources on the basis of business process models. The discipline that is concerned by this process-centric trend is known as Business Process Management (BPM) [15].

In Business Process Management the objective of the Business Process modeling phase is to capture the behavioural aspects (i.e., how to do it) of a certain business goal into a business process model [14]. There are several modeling approaches that can be split in two categories. The first one consists of designing business process models from scratch, which is an error prone and time consuming task [5]. The second category relies on reusing existing business process models.

The advent of Reuse-Oriented Development (ROD) in BPM brings a number frameworks used to support the design of business process models exploiting proven practices. One of these frameworks is the configurable process model.

Configurable process models are constructed via the aggregation of several *variants* of a process model [12]. In fact, under different requirements, different business processes could achieve the same business goal. We call these business processes business process *variants*. Since they model in essence the same business goal, these variants share many commonalities. Therefore, managing these variants can be made easier by handling the common parts just once and not for each variant separately. In order to provide easier maintenance and management of variants, a key aspect of variability handling in process modeling is the explicit representation of *variation points*. A variation point is a special placeholder in the configurable process model in which variants are defined. During the business process modeling phase, the configurable process model is configured by setting up the variation points according to a user's specific requirements. These variation points capture different requirements that discriminate between the distinct parts of business process variants through *configuration parameters*.

To manage a configurable process model, we propose an indexing structure that captures variability at the business goal level. In this paper we define a hierarchical representation of configurable process models where a variation point is a business goal that has more than one business variant to achieve it. The rationale we opt for a hierarchical structure, which explicitly captures variation points, is to provide a user-friendly experience during the modeling phase while not overwhelming the modeler with cumbersome details from start.

Most of recent studies [1,3,4,6,10,11,12] consider managing business process variability under a single governance. However, in an open environment, managing variability with a top-down approach is not suitable. We propose an indexing structure that can be applied in such environments as it allows for defining new variation points by just adding a new variant of any business goal.

Our structure allows also for modularity handling. By modularity we mean the possibility to configure a sub process within the configurable process model. This is not possible with current approaches as the configuration phase consists of parsing the whole configuration process model.

The remainder of the paper is structured as follows. Section 2 describes a use case scenario to motivate the use of configurable process models in business process management. The indexing structure as well as its construction principle are presented in Section 3. Section 4 discusses some related work while Section 5 concludes the paper.

2 Motivating Example

In this section we give a motivating example for configuration-based modeling. The example describes a configurable process model in the job recruiting area as depicted in Fig. 1[1]. The process variability is modeled through a hierarchical structure of business goals (i.e., 'Find Possible Job Candidates' , 'Evaluate

[1] All figures in the paper are following the BPM notation: www.bpmn.org

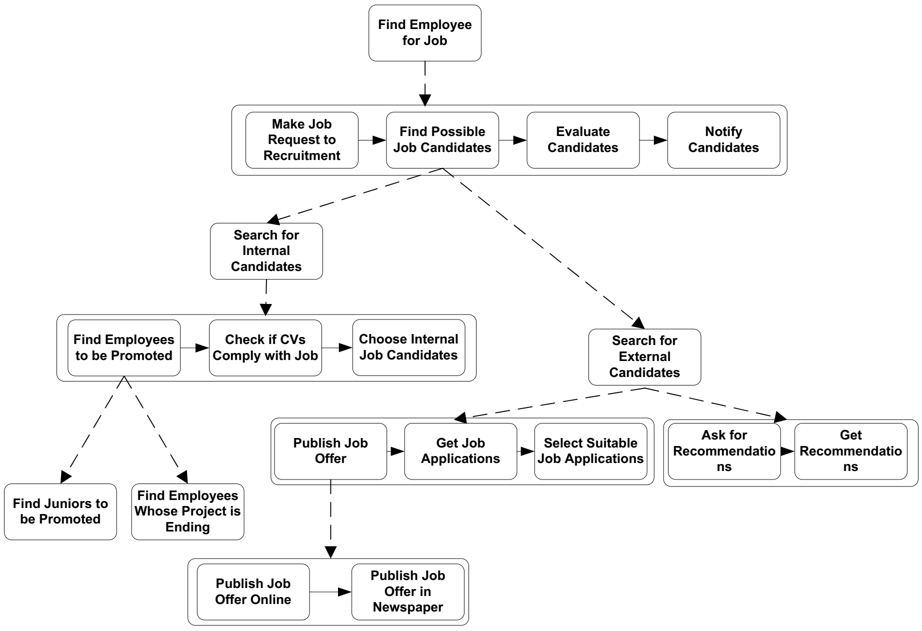
Find Employee for Job
Make Job Request to Recruitment
Find Possible Job Candidates
Evaluate Candidates
Notify Candidates
Search for Internal Candidates
Find Employees to be Promoted
Check if CVs Comply with Job
Choose Internal Job Candidates
Search for External Candidates
Find Juniors to be Promoted
Find Employees Whose Project is Ending
Publish Job Offer
Get Job Applications
Select Suitable Job Applications
Ask for Recommendatio ns
Get Recommendatio ns
Publish Job Offer Online
Publish Job Offer in Newspaper

Fig. 1. The configuration process model from the job recruitment area

Candidates', 'Notify Candidates' etc.). In our model, we capture generalization-specialization relations between business goals. For example the 'Search for Internal Candidates' goal can be achieved at a lower layer of detail by a composed goal which integrates the 'Find Employees to be Promoted', 'Check if CVs Comply With Job' and 'Choose Internal Job Candidates' business goals in a sequence block pattern.

Variation points are modeled as business goals which can be achieved by many goals (simple or composite) one layer below in the hierarchy. This is represented in the Fig. 1 through the dotted line. For example, the 'Search for External Candidates' goal is a variation point because there are different ways of being achieved, either by publishing a job offer or by getting recommendations from acquaintances.

A typical scenario in this context is to find an employee in an enterprise for an available job, represented by the 'Find Employee for Job' business goal. We notice that there are many variants of such a process. At a certain moment, a business user would prefer a process that searches for internal job candidates(1), while in another context he would choose an external search(2). Normally, the business user would explore the configuration process model and, at the variation points, he would choose the most satisfiable alternative in a given context. Fig. 2 shows two possible variants.

Let us imagine that the configuration process model does not contain configuration parameters yet. In this situation configuration-based modeling is a

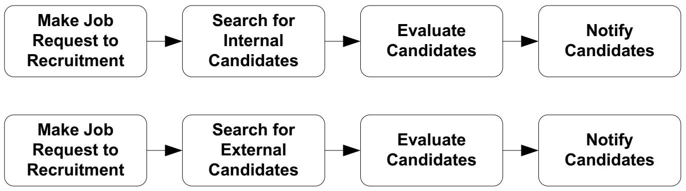
Make Job Request to Recruitment
Search for Internal Candidates
Evaluate Candidates
Notify Candidates
Make Job Request to Recruitment
Search for External Candidates
Evaluate Candidates
Notify Candidates

Fig. 2. Two different variants of 'Find Employee for Job' goal

cumbersome task for the business user because distinctive aspects are not captured in a user-friendly way in order to support the business user to make his selection.

That is why the need for well structured information represented by configuration parameters at variation points arises. Configuration parameters of the variation points represent a set of functional and nonfunctional aspects of all possible alternatives. For example, the configuration parameters of the 'Find Possible Job Candidates' business goal are the *time* with the options :[1-2 weeks] or [3-4 weeks] (non functional aspects) and the *candidateType* with the options: [internal] or [external](functional aspects).

In this paper we do not focus on presenting configuration parameters but rather on managing a data structure to capture the hierarchical representation of this goal hierarchy. The next section will introduce our reference model indexing structure.

3 Indexing Structure for Model Variations

In this section we define an indexing structure to represent configurable process models. We give a set of structuring principles and we show how to maintain this structure when adding a new variant. In our study we consider business goals as the main artifact of the configurable process model. We introduced the concept *abstract business goal* for manipulation purposes. A business process model is represented by at least one concrete business goal (i.e., it can be a sequence of business goals without any abstract one).

3.1 Indexing Structure Definition

We define a data structure to maintain configurable business process models. A configurable business process model is a tuple CPM=$\{\Sigma,\Gamma,\Delta\}$ where:

- Σ represents the set of business goals involved in the whole reference business process model,
- Γ represents the set of abstract business goals (will be introduced later in this paper) and
- Δ represents the possible variants of each business goal that can be a sequence of business goals. Entries of Δ are presented as:

$BusinessGoal_1 : BusinessGoal_2(-BusinessGoal_N)^*$ such that $\{BusinessGoal_1, BusinessGoal_2, ...BusinessGoal_N\} \subseteq \Sigma \cup \Gamma$ and "$BusinessGoal_2 - BusinessGoal_3$" is the sequence between the two business goals.
This means that possible variants of BusinessGoal1 are presented as a sequence of other business goals. We call $BusinessGoal_1$ a *variation point* and $BusinessGoal_2(-BusinessGoal_N)^*$ a *variant.*

As an illustration, the motivating example of Fig. 1 would be presented as follows:

$CPM_1 = \{\Sigma_1, \Gamma_1, \Delta_1\}$ where:

- Σ_1 = { [Find Employee for Job], [Make Job Request to Recruitment], [Find Possible Job Candidates], [Evaluate Candidates], [Notify Candidates],[Search for Internal Candidates], [Find Employees to be Promoted], [Check if CVs Comply with Job], [Choose Internal Job Candidates], [Find Juniors to be Promoted], [Find Employees Whose Project is Ending], [Search for External Candidates], [Publish Job Offer], [Get Job Applications], [Select Suitable Job Applications], [Publish Job Offer Online], [Publish Job Offer in Newspaper], [Ask for Recommendations], [Get Recommendations] }
- Γ_1 ={ }
- Δ_1 = { [Find Employee for Job] : [Make Job Request to Recruitment] - [Find Possible Job Candidates] - [Evaluate Candidates] - [Notify Candidates],
  [Find Possible Job Candidates] : [Search for Internal Candidates],
  [Find Possible Job Candidates] : [Search for External Candidates],
  [Search for Internal Candidates] : [Find Employees to be Promoted] - [Check if CVs Comply with Job] - [Choose Internal Job Candidates],
  [Find Employees to be Promoted] : [Find Juniors to be Promoted],
  [Find Employees to be Promoted] : [Find Employees Whose Project is Ending],
  [Search for External Candidates] : [Publish Job Offer] - [Get Job Applications] - [Select Suitable Job Applications],
  [Publish Job Offer] : [Publish Job Offer Online],
  [Publish Job Offer] : [Publish Job Offer in Newspaper],
  [Search for External Candidates] : [Ask for Recommendations] - [Get Recommendations]}

As it was shown in Fig. 2, both variants for 'Find Employee for Job' can be generated from this structure. It allows also for managing modularity as we can configure even modules like 'Search for External Candidates' that has two variants. In the rest of the paper, for simplicity, we will use letter (e.g., A, B, C...) as labels for the business goals in order to avoid long strings.

3.2 Construction Principles

The indexing structure should respect a set of constraints/principles that assure it will remain valid and well-formed even after any update operation (i.e. adding a new variant).

We have identifies three principles: Minimality, Coverage and Consistency. They are presented here with a definition and an example.

Minimality

Definition: Each element of Δ has to be defined only once and should not be derived from other elements of Δ.

Example: $CPM_1 = \{\Sigma_1, \Gamma_1, \Delta_1\}$ where:

- $\Sigma_1 = \{$A, B, C, D, E, F$\}$
- $\Gamma_1 = \{ \}$
- $\Delta_1 = \{$A:B-C-D, A:E-F-C-D, B:E-F$\}$

Being minimal imposes that Δ_1 represents the optimal representation of alternatives construction. With respect to this constraint, $\Delta 1$ should be $\Delta_1 = \{$A:B-C-D, B:E-F$\}$ (see Fig. 3).

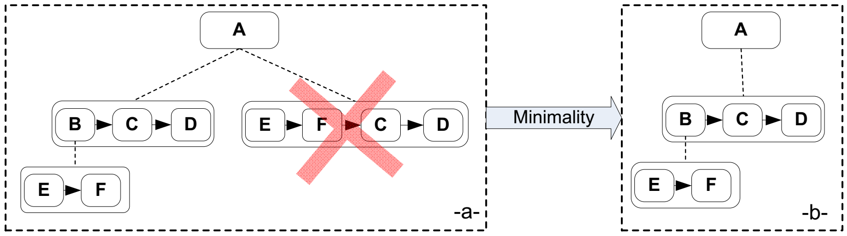

Fig. 3. The minimality principle

Coverage

Definition: : By necessity and nature, the indexing structure must cover all possible variants of the configurable business process model.

Example: In the example of Fig. 4 we can notice that possible variants generated from Δ (i.e., Fig. 4.a) into Fig. 4.b do not cover all possible variants from the Fig. 4.c.

Consistency

Definition: : Only defined variants should be deduced from the indexing structure and no extra ones are allowed to appear.

Example: : Considering that we have the list of defined variants in Fig. 5.a, the tree from the Fig. 5.b is wrong with respect to the set of defined variants. In fact we can deduce from it an extra variant which did not exist initially (e.g. B-G-H). The tree depicted in Fig. 5.c is a consistent structure.

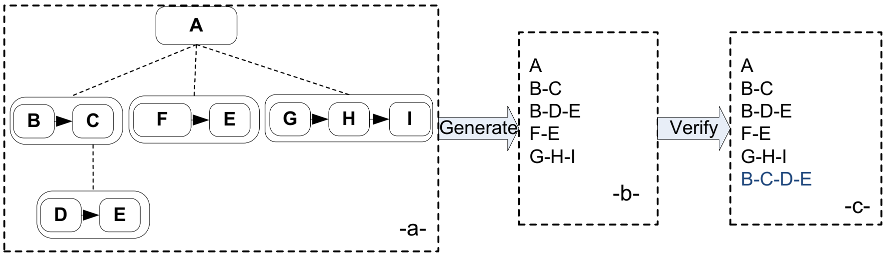

Fig. 4. The coverage principle

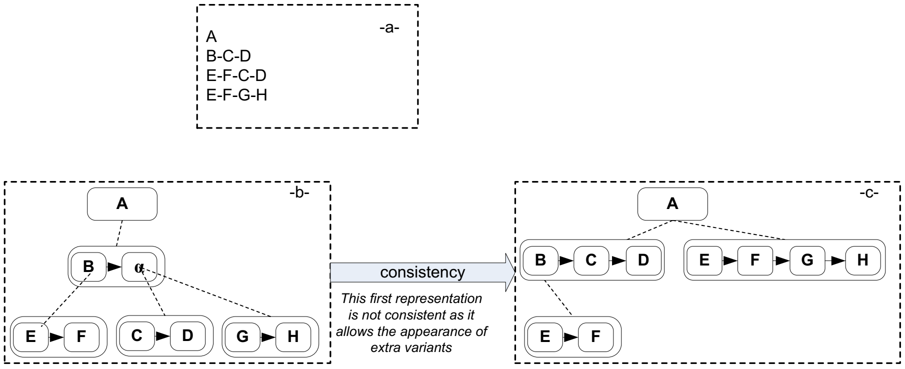
-a-
A
B-C-D
E-F-C-D
E-F-G-H
-b-
A
B
α
E
F
C
D
G
H
consistency
This first representation is not consistent as it allows the appearance of extra variants
-c-
A
B
C
D
E
F
G
H
E
F

Fig. 5. The consistency principle

3.3 Maintaining the Indexing Structure When Adding a New Variant

When building the indexing structure, we start from a set of variants and we add them, one at a time, in the current structure while assuring the principles defined previously (see section 3.2) are not violated. When adding a new variant of a goal, the idea is to check, using matching detection, whether the variant (or parts of it) already exists in the indexing structure. There are three situations that may occur according to the matching degree between business goals of the new variant and those in the configurable business process model.

1. *Perfect match:* In this case the current variant to be inserted is entirely found in the current data structure. In such situation there is no action to be taken and the data structure remains as it.
2. *No matching:* In this case the current variant to be inserted is not found in the current data structure, not even partially. The variant is inserted as follows: all business goals composing the variant are added to Σ and the variant description is added to Δ.

Example: We want to insert the variant A:G-H-I in $CPM_1 = \{\Sigma_1, \Gamma_1, \Delta_1\}$ where:

- Σ_1 = {A, B, C, D, E, F}
- Γ_1 ={ }
- Δ_1 = {A:B-C, A:F-E, C:D-E}

The updated reference process model is then CPM_1={Σ_1,Γ_1,Δ_1} where:

- Σ_1 = {A, B, C, D, E, F, G,H,I}
- Γ_1 ={ }
- Δ_1 = {A:B-C, A:F-E, C:D-E, A:G-H-I}

Using a tree representation, the variant is inserted as an alternative of the variation point as shown in Fig. 6[2].

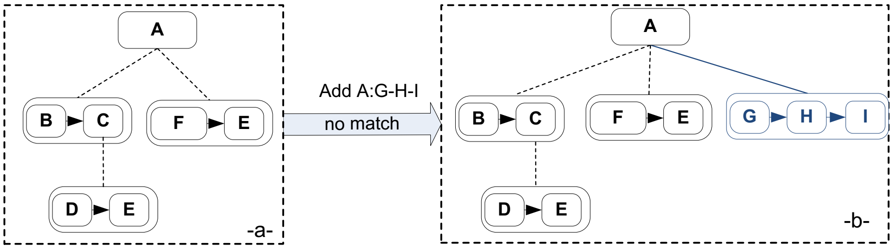

Fig. 6. The data structure (-a-) before and (-b-) after the insertion of a variant

3. *Partial match:* An intermediary situation is when a partial match occurs between the process variant to be inserted and the current data structure. In this case we distinguish two possible situations:

   - The first situation occurs when the new variant has common parts with another variant of the same business goal. A typical example is depicted in Fig. 7. This example shows adding the variant A:B-H-I in $CPM_1 = \{\Sigma_1, \Gamma_1, \Delta_1\}$ where:

     - Σ_1 = {A, B, C, D, E, F, G}
     - Γ_1 ={ }
     - Δ_1 = {A:B-C, A:F-G, C:D-E}

     The new variant A:B-H-I has B in common with A:B-C. To add this variant, an abstract business goal α is introduced to replace the different parts of these variants. The updated reference process model is then CPM_1={Σ_1,Γ_1,Δ_1} where:

     - Σ_1 = {A, B, C, D, E, F, G, H, I}
     - Γ_1 ={ α}
     - Δ_1 = {A:B-α, α:C, α:H-I,A:F-G, C:D-E}

[2] All figures in the paper are following the BPM notation: www.bpmn.org

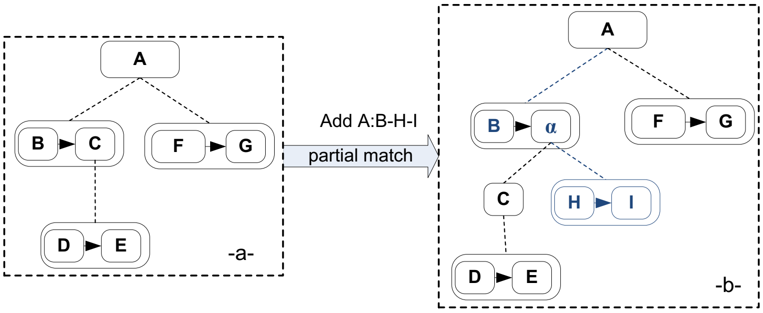

Fig. 7. The insertion of a variant including an abstract business goal

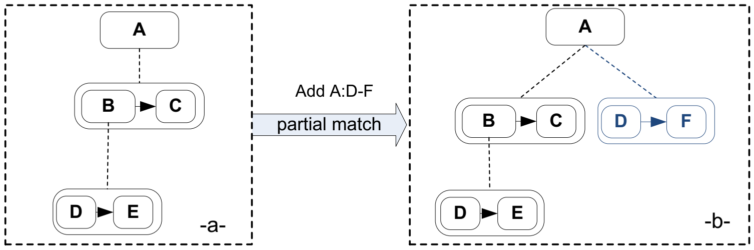

Fig. 8. The insertion of a variant with partial match without introducing an abstract business goal

– The second situation occurs when the new variant has common parts with another variant but not of the same business goal. A typical example is depicted in Fig. 8. This example shows adding the variant A:D-F in $CPM_1 = \{\Sigma_1,\Gamma_1,\Delta_1\}$ where:
  - $\Sigma_1 = \{$A, B, C, D, E$\}$
  - $\Gamma_1 = \{\ \}$
  - $\Delta_1 = \{$A:B-C, B:D-E$\}$

  This situation is similar to the second case (no match) and the updated reference process model is then $CPM_1 = \{\Sigma_1,\Gamma_1,\Delta_1\}$ where:

  - $\Sigma_1 = \{$A, B, C, D, E, F$\}$
  - $\Gamma_1 = \{\}$
  - $\Delta_1 = \{$A:B-C, B:D-E,A:D-F$\}$

4 Related Work

Several approaches have been proposed for defining and managing business process variants. In this section we state four current approaches dealing with process variability.

The first approach is the most intuitive solution to variability management. It consists of managing a repository of process variants. Each process model is stored as an individual entity in the repository. Users have to formulate a query according to their requirements and the system should provide the most suitable model. This approach has been explored by [7,8,9] where it reveals that it needs a rich formal model for describing business process. In our work, we do not use individual models because the main problems of this solution are resource allocation and inconsistency. Indeed, (i) storing each variant individually leads to duplicated data storage for common parts of the process models and (ii) in case of new regulations enforcement, all process variants have to be updated which is resource consuming and error prone task. In addition, variation points are not explicitly handled and in [7], configuration-based modeling relies on querying the process models repository based on structural aspects of the to-be process. Therefore the business user has to know what are the possible process structures he is allowed to ask for.

The second approach as it is presented in [4,3], overcomes the problems of resource allocation and inconsistency. This solution considers a "basic process model" that represents common parts of all process models and variability is handled as a global property containing a set of operations (e.g., add, delete, modify, move operation). In fact, each variant is then generated via applying these operations on the basic model. However, the business user's control becomes limited to a set of operations generating rules which fire when they comply with all the business requirements. These rules capture only non functional aspects (i.e., quality aspects like cost and performance) leaving out details about structural and functional aspects of the variants.

The third approach consists of generating a global flat process model containing all variations and each individual model is generated by eliminating some branches of the global model. [1,6] model process variability as explicit variation points within the control structure of a flat configurable model. However this solution poses visualization problems because, in a real world setting with a lot of process variants, the configurable process tends to get very large. Therefore the configuration model becomes difficult to comprehend and costly to maintain. But [11,12] reduced these problems by presenting a questionnaire-based configuration which is much more user-friendly than previous solutions.

In [11,12], the user specifies his business requirements by answering a set of domain-related questions. The authors distinguish between domain variability (i.e., it is based on domain facts which are features that can be enabled or disabled) and process variability (i.e., it is based on possible alternatives at a certain variation point). Both are related through a set of mappings such that the result of the domain-specific questions are reflected in the chosen alternative for a variation point. It is a very good option to make configuration user centric but IT experts are still highly needed to define both domain and model variability and their mapping which is manually performed and this makes the approach liable to subjectivity.

In addition, this approach is not flexible enough to manage modularity. Indeed, if the user wants to configure a particular business function that is embedded in the global configurable model, he has to go through this model until reaching the intended business function to be configured. In our solution this problem cannot occur because we consider individual entities that can range from simple activities to complete process models.

The fourth approach studied in [10], is similar to ours as it exploits a hierarchical representation of the process into sub processes. The top level sub process encompasses the core activities and their associated variability, which is annotated by specific stereotypes, while the lower level sub processes express all details related to higher level activities and variabilities residing in them. However, the concept of hierarchical representation is supported more for hiding complexity than for managing variability.

5 Conclusion and Future Work

Configuration-based modeling is an important approach for business process management because it can decrease the modeling time and reduce the business user's work and risk to make errors. Configuration-based modeling is an iterative process of refinement actions in which the business user is assisted to lookup the most suitable process model depending on his requirements. Reviewing approaches that deal with configuration-based modeling, we have determined that they manage business process variability under a single governance as well as that they do not support modularity.

In this paper we proposed a structure for managing configurable process models. It is defined as a hierarchical indexing structure that captures process model's variability at the business goal level. We present a set of principles that the proposed indexing structure has to comply with and we show how it is maintained when adding a new variant to the configurable process model.

Our work is still in progress and continuous improvements are planned as a future work:

- We plan to formally define construction principles. New ones could be defined as well.
- A number of maintaining operations have not been yet explored or are still under definition, for example the deletion of a variant.
- We intend to investigate and extend this indexing structure in order to provide support for other block patterns (we have presented only sequence pattern in this paper).
- Our indexing structure is not exclusively designed for managing process variability. We eventually would experiment it in Mashups development environment to capture variability within Mashup applications.

Acknowledgment

This work is founded by the Lion II project supported by Science Foundation Ireland under grant number 08/CE/I1380.

References

1. Dreiling, A., Rosemann, M., van der Aalst, W.M.P., Sadiq, W., Khan, S.: Model-driven process configuration of enterprise systems. In: Ferstl, O.K., Sinz, E.J., Eckert, S., Isselhorst, T. (eds.) Wirtschaftsinformatik, pp. 687–706. Physica-Verlag, Heidelberg (2005)
2. Dumas, M., van der Aalst, W.M., ter Hofstede, A.H.: Process-Aware Information Systems: Bridging People and Software Through Process Technology. Wiley-Interscience, Hoboken (2005), ISBN: 0471663069
3. Hallerbach, A., Bauer, T., Reichert, M.: Context-based configuration of process variants. In: Proceedings of the 3rd International Workshop on Technologies for Context-Aware Business Process Management, TCoB (2008)
4. Hallerbach, A., Bauer, T., Reichert, M.: Managing process variants in the process life cycle. In: Cordeiro, J., Filipe, J. (eds.) ICEIS (3-2), pp. 154–161 (2008)
5. Indulska, M., Green, P., Recker, J., Rosemann, M.: Business process modeling: Perceived benefits. In: Laender, A.H.F., Castano, S., Dayal, U., Casati, F., de Oliveira, J.P.M. (eds.) ER 2009. LNCS, vol. 5829, pp. 458–471. Springer, Heidelberg (2009)
6. Lapouchnian, A., Yu, Y., Mylopoulos, J.: Requirements-driven design and configuration management of business processes. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 246–261. Springer, Heidelberg (2007)
7. Lu, R., Sadiq, S.W.: On the discovery of preferred work practice through business process variants. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, pp. 165–180. Springer, Heidelberg (2007)
8. Markovic, I., Pereira, A.C.: Towards a formal framework for reuse in business process modeling. In: ter Hofstede, A.H.M., et al. (eds.) [13], pp. 484–495
9. Markovic, I., Pereira, A.C., Stojanovic, N.: A framework for querying in business process modelling. In: Bichler, M., Hess, T., Krcmar, H., Lechner, U., Matthes, F., Picot, A., Speitkamp, B., Wolf, P. (eds.) Multikonferenz Wirtschaftsinformatik. GITO-Verlag, Berlin (2008)
10. Razavian, M., Khosravi, R.: Modeling variability in business process models using uml. In: ITNG, pp. 82–87. IEEE Computer Society, Los Alamitos (2008)
11. Rosa, M.L., Lux, J., Seidel, S., Dumas, M., ter Hofstede, A.H.M.: Questionnaire-driven configuration of reference process models. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 424–438. Springer, Heidelberg (2007)
12. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. Inf. Syst., 32(1):1–23 (2007)
13. ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.): BPM Workshops 2007. LNCS, vol. 4928, pp. 66–77. Springer, Heidelberg (2008)
14. van der Aalst, W.M.P.: Business process management. In: Liu, L., Özsu, M.T. (eds.) Encyclopedia of Database Systems, pp. 289–293. Springer, US (2009)
15. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Springer, Heidelberg (2007)