Automatic Generation of Questionnaires for Supporting Users during the Execution of Declarative Business Process Models

Andrés Jiménez-Ramírez, Irene Barba, Barbara Weber, and Carmelo Del Valle

Abstract. When designing an imperative business process (BP) model, analysts have to face many design requirements (e.g., managing uncertainty, optimizing conflicting objective functions). To facilitate such design, declarative BP models are increasingly used. However, how to execute a given declarative model can be quite challenging since there are typically several variants related to such model, each one presenting different degree of goodness. To support users working on declarative models while a high flexibility is maintained, we propose removing the worst variants from the source declarative model at design time while keeping the best variants. This way, the variants which are kept are narrowed down incrementally during run-time. For managing these variants during run-time we suggest to build upon configurable BP models. To configure such models, we additionally propose to automatically generate questionnaires. The results over a real case study are promising.

Keywords: Declarative Business Process Models, Configurable Business Process Models, Questionnaires.

1 Introduction

Business Process (BP) models are typically specified by hand using imperative languages like EPC or BPMN [3]. When designing an imperative BP model, analysts have to face many design requirements (e.g., dealing with activity attributes, managing uncertainty, dealing with relations between activities, considering the optimization of potentially conflicting objective functions, etc. [11]). To facilitate the human work involved in such design, to avoid failures, and to allow for a better optimization during the execution phase [7], declarative BP models are increasingly used allowing their users to specify what has to be done instead of how [1,10]. However, due to their flexible nature, there are frequently several variants related to a given declarative model, each one presenting specific values for different objective functions (e.g., overall completion time or profit). Therefore, the decision about how to execute this declarative model (i.e., selecting a variant that finally gets executed) can be quite challenging.

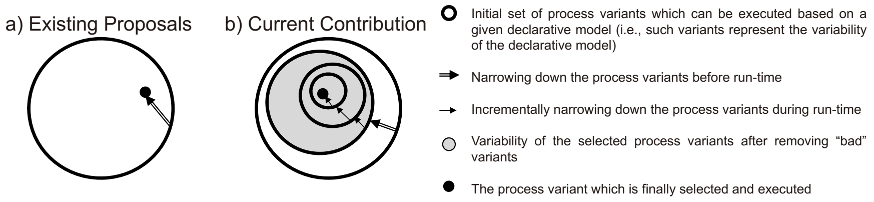

a) Existing Proposals
b) Current Contribution
Initial set of process variants which can be executed based on a given declarative model (i.e., such variants represent the variability of the declarative model)
Narrowing down the process variants before run-time
Incrementally narrowing down the process variants during run-time
Variability of the selected process variants after removing "bad" variants
The process variant which is finally selected and executed

Fig. 1. Current contribution VS existing proposals

For supporting users working on declarative BP models, we proposed in previous works [2,1] an approach for generating an optimized execution plan (i.e., an optimized variant) from a given declarative BP model at design time (cf. Fig. 1 (a)). However, as a major disadvantage of such previous work, only one single variant is selected before starting the process execution which unnecessarily restricts the flexibility (cf. black dot in Fig. 1 (a)) [1,10], and hence, diminishes the advantages of using declarative models. In particular, if BPs are subject to uncertainty and conditions may change during the BP execution, it might turn out that the selected variant is not applicable and replanning might be required. To be better able to cope with such uncertainty, it is more suitable to defer the decisions of how the variant to be executed looks like to run-time. To be more specific, instead of narrowing down the selection to one single variant before run-time (cf. Fig. 1 (a)), it would be better that only *the worst* variants are removed while the *the best* variants are kept (cf. the outermost gray circle in Fig. 1 (b)). Thereby the goodness of a variant is measured by its values for given objective functions [10]. This way, the variants which are kept can be narrowed down incrementally during run-time at the last possible moment (i.e., gradually moving from the outermost inner circle to the back dot in Fig. 1 (b)).

To support users working on declarative BP models while a high flexibility is maintained, we propose unlike [2,1] to not select a single variant, but to keep the best variants before the BP enactment. For managing these variants during run-time, they can be automatically merged into a configurable BP (CBP) model (i.e., a modeling artifact that captures families of BP models in an integrated manner [16,15]) by using the techniques presented in [9]. Such a model then allows analysts to understand what these variants share, what their differences are, and why and how these differences occur.

To configure CBP models in such a way that domain experts incrementally reduce the number of process variants to be executed, we additionally propose to automatically generate questionnaires (i.e., sequences of questions each one created for configuring a part of a CBP model [12]). While the usage of questionnaires is not new [12,13], existing works require that analysts manually create the questionnaires, unlike the current proposal. In addition, such a configuration is done at configuration time (i.e., before process execution starts), and hence, unlike the current proposal and other proposals as aspect-oriented approaches [6], premature decisions may unnecessarily be taken.

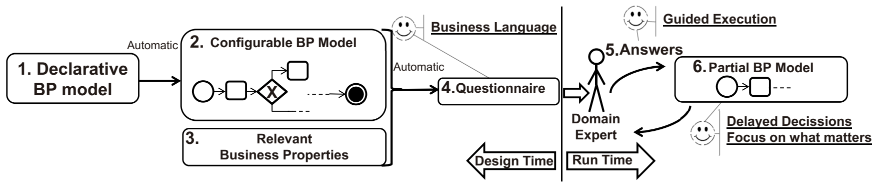


Fig. 2. Overview of our contribution

The first part of the current contribution (i.e., the generation of the best variants to be kept from a declarative BP model and the creation of a CBP model out of them) has been already presented in previous works [9,10]. However, this paper significantly extends [9,10] by: (1) Automatically generating the questionnaires for configuring the CBP model and (2) incrementally configuring the CBP model during run-time using the generated questionnaire.

As depicted in Fig. 2, in our proposal, a declarative BP model (cf. Fig. 2 (1)) is used as starting point. Then a CBP model is automatically generated out of it (cf. Fig. 2 (2)) by selecting the best variants as detailed in [9,10]. Then, using the generated CBP model together with a set of well-defined relevant business properties (i.e., properties that can be measured within each variant and which are understandable by the domain expert, cf. Fig. 2 (3)), we automatically generate a questionnaire without involving the analyst. Such a questionnaire consists of questions written in the business language (cf. Fig. 2 (4)). Thereafter, the domain expert interacts with the questionnaire to configure the CBP model herself during run-time. This way, the generated questionnaire allows to narrow down the variants of the CBP model in an incremental way during run-time, i.e., guiding the execution of the CBP model by answering the questionnaire (cf. Fig. 2 (5)). Therefore, the BP model is partially created (cf. Fig. 2 (6)) and thus, it is possible to execute already configured parts. In addition, as users often do not have an understanding of the overall process, they can focus only on the part of the CBP model to be configured, which may help them to take decisions.

Note that our approach is appropriate for managing scenarios which present certain requirements, i.e., scenarios which (1) are subject to changes (e.g., company best practices which change due to the customers feedback), (2) have a well-defined set of business properties which can be extracted for a variant (e.g., the property 'completion time' of a variant can be related to the 'opening and closing time' of the business), and (3) highly rely on domain expert's skills (i.e., decisions influence business performance) and thus, decisions can not be predefined. As an example of such a scenario, the suitability of the current proposal has been validated through a real scenario. Nevertheless, the proposed approach is not restricted to such scenarios, but can be applied over any CBP model.

This paper is organized as follows: Sect. 2 introduces backgrounds on related areas, Sect. 3 details the proposed method, Sect. 4 deals with the evaluation, and Sect. 5 includes some conclusions and future work.

2 Background

Declarative Models: Different paradigms for process modelling exist, e.g., imperative [3] and declarative [7]. Imperative process models are well structured representations which specify exactly how things have to be done by explicitly depicting all possible behavior. A declarative model, in turn, is a loosely-structured representation focused on what should be done restricting all forbidden behavior. Therefore, declarative models are commonly used for representing processes with high variability which can be executed in several ways (cf. Example 1). In the context of declarative models, a constraint-based model can be defined as a set of activities which can be executed following a given set of behavioural constraints (e.g., resource and control flow constraints).

Example 1. Figure 3 (a) shows a constraint-based BP model together with some valid and invalid traces[1]. In contrast, Fig. 3 (b) shows an imperative model where there is only one valid execution trace.

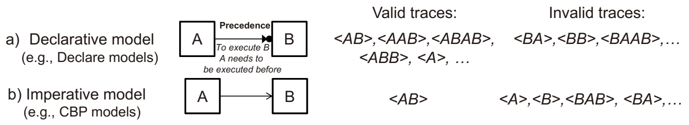


Fig. 3. Increased flexibility of declarative models versus imperative models

Due to their flexible nature, there are frequently different ways to execute a constraint-based model in such a way that all constraints are fulfilled. The different valid execution alternatives (i.e., variants), however, can vary significantly in how well different performance objective functions (e.g., benefit and time) can be achieved. For generating variants from a constraint-based model optimizing given objective functions, we applied planning & scheduling techniques in previous work [10]. Each variant which is generated can be represented as a BP graph (cf. Def. 1).

Definition 1. *A* ***BP Graph*** $G = (gid, N, Pairs)$ *is identified by gid and consists of a set of pairs of nodes* $n \in N$*, i.e.,* $Pairs$*. Each pair denotes a direct edge between two nodes in the graph. A node* $n \in N$ *is a tuple* $< nid, l, t >$ *where nid is an unique identifier of a node in the graph, l is its label, and t is its type (e.g., activities, events, and gateways).*

Such definition of graph allows to represent a BP model in many different imperative BP languages [3], e.g., BPMN or EPC. As an example, the types of nodes (i.e., t) in BPMN language [3] are 'activity', 'event', or 'gateway'. A

[1] For the sake of clarity, traces represent sequences of activities, i.e., no parallelism is considered in the examples. Moreover, only completed events for activity executions are included in the trace representation.

node of type 'gateway' allows the labels (i.e., l) 'AND', 'OR', 'XOR', etc., while 'event' nodes allow 'start' and 'end' labels.

Configurable BP Models: Typically, different variants can be performed in scenarios which entail high variability. In most cases these variants share many commonalities, and hence, can be combined in a CBP model leading to a compact representation [15,16,12]. CBP models are typically created by hand (1) from scratch, (2) from an existing BP model by including possible adaptations [8], or (3) by merging some BP models related to the same or similar goals which already exist [15]. In the latter case, there exist approaches focused on automatically merging different BP models into a CBP model [14,15].

In a similar way to each variant can be represented as a BP graph (cf. Def. 1), CBP models can be represented as CBP graphs, which are defined as described in [15] (cf. Def. 2).

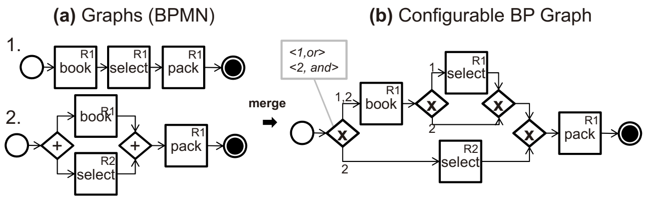


Fig. 4. Two BP graphs (a) are merged into a single configurable BP graph (b)

Definition 2. *A* ***Configurable BP Graph*** $CG = (G, E2I, N2LI)$ *consists of: (1) a BP graph,* $G = (gid, N, Pairs)$ *(cf. Def. 1), (2) a function* $E2I$ *that maps each edge* $e \in Pairs$ *to a set of BP graph identifiers (i.e.,* $E2I$ *states which branches of* CG *belong to each source BP graph which is merged in* CG*), (3) a function,* $N2LI$ *that maps each node* $n \in N$ *to a set of pairs* $< gpid, l >$ *where* $gpid$ *is a BP graph identifier and* l *is the label of the node* n *in the graph* $gpid$ *(i.e.,* $N2LI$ *states which nodes, with the corresponding label, belong to each source BP graph which is merged in* CG*).*

A CBP graph includes *configuration nodes* for those points where the BP graphs which are merged differ (cf. Example 2). Therefore, each branch and each node of the CBP graph can be related either to one or more BP graphs. As mentioned in Def. 2, to store these relations, each branch/node of the configurable BP graph includes identifiers related to the corresponding BP graph (i.e., $E2I$ function). In addition, the nodes of the CBP graph also store the associated label related to each identifier (i.e., $N2LI$ function).

Example 2. Figure 4 shows 2 BP graphs which are merged into a CBP graph [2]. As can be observed, the first gateway in Fig. 4(b) is a configurable node which

[2] As there is not ambiguity, some labels are not shown (i.e., they are the same as in the branch).

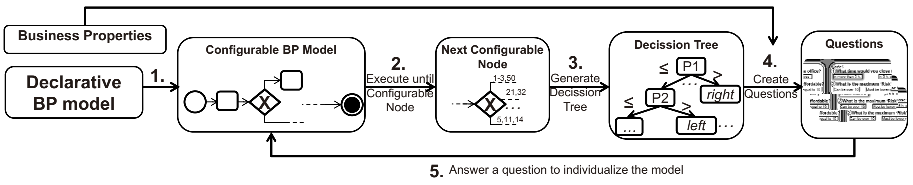


Fig. 5. Automatic generation of questionnaires for Individualizing a CBP model

corresponds to an 'OR' gateway in BP Graph 1 (it does not explicitly appear) and an 'AND' gateway in BP Graph 2.

Questionnaire Models: Questionnaire models [13] are typically created by the analysts to support the user during the configuration (i.e., individualization) of the CBP models. The main benefits of using them are: (1) they guide the user in such a way that choices are presented in a proper order and (2) they avoid invalid configurations which may lead to errors.

Typically, questionnaires are manually created by an analyst whereby each question is related to boolean *facts* which are associated to configuration *actions* [13]. Each time a question is answered, an action which configures a part of the CBP model is fired. The sequence of answers given to the different questions will individualize the CBP model in such a way that a single variant is selected before run-time to be executed.

Unlike previous approaches which deal with questionnaires, this work: (1) automatically creates the questionnaires (i.e., defining facts and actions are not longer needed) and (2) the questionnaires which are created are intended to individualize the CBP models during run-time (cf. Sect. 3).

3 Individualizing a CBP Model through the Automatic Generation of Questionnaires

In this section, our method for automatically generating questionnaires from a declarative model and its usage for supporting the user during the execution of such model is described (cf. Fig. 5). As a first step, a CBP model is generated out of the source declarative model (cf. Fig. 5 (1)) as detailed in [9,10]. Then, the BP execution starts and advances until a configurable node (cf. Def. 2) is found in the CBP model (cf. Sect. 3.1, Fig. 5 (2)). Thereafter, a decision tree related to such configurable node is created (cf. Sect. 3.2, Fig. 5 (3)) as an intermediate step for generating the questionnaire associated to this configurable node (cf. Sect. 3.3, Fig. 5 (4)). Whenever the user answers a questionnaire (i.e., a decision is taken, cf. Sect. 3.4, Fig. 5 (5)), the variants of the CBP model are narrowed down based on the answers given. This method is iteratively applied from step 2 to step 5 until no more individualization is needed (i.e., until only one single variant remains in the CBP model).

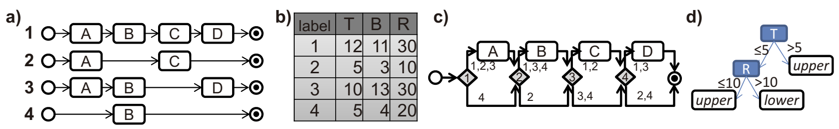
3.1 Executing the Configurable BP Model

As stated in Def. 2, all variants which are included in the CBP model are labeled (cf. Example 3).

Example 3. The running example of Fig. 6(a) comprises four BP models, each one labeled with an integer. Furthermore, a group of properties for each BP model is provided (cf. Fig. 6 (b) where time (T), benefit (B) and risk (R) properties are provided for each model). Such properties are related to the business language, e.g., T is related to the opening hours of the business. The CBP model associated with the BP models which are depicted in Fig. 6 (a) is shown in Fig. 6 (c). In this model, 4 different configurable nodes are depicted with a bold diamond. In the first configurable node, labeled as 1, two alternatives are possible. The *lower* branch comprises BP Model 4 (i.e., where activity A is not executed), and the *upper* branch comprises BP Models 1 to 3 (where activity A is executed).

The CBP model can be executed from the beginning until a configurable node appears, i.e., until a decision must be taken (cf. Fig. 5 (2)).

Note that the selection of a valid variant is guaranteed since we are building upon our previous work which generates valid variants. Merging them preserves these variants and the same happens with the decision trees.



b)

label	T	B	R
1	12	11	30
2	5	3	10
3	10	13	30
4	5	4	20

Fig. 6. a) 4 different BP models. (b) Properties of each BP model. (c) CBP model related to the BP models of (a). (d) Classification tree for node 1.

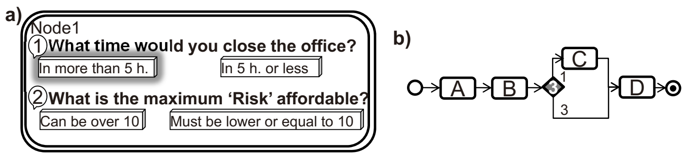

a)
Node1
① What time would you close the office?
In more than 5 h. | In 5 h. or less
② What is the maximum 'Risk' affordable?
Can be over 10 | Must be lower or equal to 10

Fig. 7. (a) Questionnaire for Node 1. (b) The resulting configurable model after removing Variants 2 and 4.

3.2 Generating Decision Trees

When a configurable node is encountered we apply a method for generating a prediction system (i.e., a model that predicts the value of a target variable based on several input variables) [4] for predicting which outgoing branch corresponds to a given assignment of property values. Specifically, for each configurable node, a classification tree is created (cf. Fig. 5 (3)) using the property values of the variants as input variables and the outgoing branches as target variables (cf. Example 4).

Example 4. Figure 6(d) shows the classification tree which comes of using the CART algorithm [4] when using the table of Fig. 6(b) as input variables and the strings *lower* and *upper* as target variables. As can be seen, in the resulting classification tree, the variants for which $T > 5$ correspond to the *upper* branch. In contrast, the variants for which $T \leq 5$ correspond to the *upper* branch if $R \leq 10$, or to the *lower* branch otherwise.

3.3 Creating Questions

A set of questions is then created for each decision tree (cf. Fig. 5 (4)). To create such questions according to the business language, a set of well-defined business properties must be provided. This way, one question is automatically generated for each intermediate node of the tree. The possible answers for such question are the different labels which are written on the outgoing branches of this node. The text of the questions is automatically generated from the information of the provided business properties (cf. Example 5). As stated, these questionnaires are in charge of narrowing down the variants of the CBP model.

Example 5. A simple questionnaire related to the decision tree of Fig. 6(d) is shown in Fig. 7(a). Since this decision tree has two intermediate nodes (i.e., T and R), two questions are created. Moreover, since each node has two branches, each question has two options. Initially, only the question related to T is enabled. Considering that the well-defined business properties stated that T is related to the *closing time* of the office, the generated question would look like *What time would you close the office?*.[3] The second question has to be answered only if the user selects the second option of the first question (i.e., *In 5hrs. or less*) which is related to the branch $T \leq 5$ of the decision tree.

3.4 Incremental Configuration

Whenever a questionnaire is answered, the CBP model is narrowed down by removing the variants that do not belong to the edge selected in this configuration step. Thereafter, the proposed method continues at Step 2 (cf. Fig. 5) considering the narrowed CBP model and continuing the execution from the last executed activity.

Such method is repeated until only one variant remains in the CBP model, i.e., the configuration has finished (cf. Example 6).

Example 6. Supposing that the user selects the first answer of the first question of the questionnaire of Fig. 7(a) (i.e., *In more than 5hrs.*), Variants 2 and 4 are removed from the CBP model since they have a time property "≤ 5". This results in the CBP model of Fig. 7(b). Note that the second and forth configurable node

[3] Note that, the semantic of the generated questions highly depends on the information provided for the business properties. Such information can be used to make the questions more user-friendly. No depth details are given since it is out of the scope of this paper.

of Fig. 6(c) are not depicted in Fig. 7(b) since Variants 1 and 3 share the same outgoing branches for these nodes, i.e., the *upper* branch. However, the third configurable node requires selecting one of the two branches, and hence, a new questionnaire is generated.

4 Case Study

This section provides and empirical study for evaluating the suitability of the proposed approach. Specifically, the case study protocol proposed by [5] is followed to improve the rigor and validity of the study.

Background: In the context of the proposed approach, the *object of study* is the method for automatically creating questionnaires for CBP models and the method for incrementally configuring them (cf. Sect. 3). In such context, the *purpose of this study* is to evaluate the suitability of both methods regarding its feasibility and effectiveness when managing a real scenario.

Considering the object and purpose of this study, a main research question is defined: ($MQ1$) Is our method appropriate for individualizing CBP models during run-time? This question is further divided into 4 additional questions: ($AQ1$) Can the proposed method be used to generate questions for configurable nodes of different sizes (i.e., nodes with different number of branches)?, ($AQ2$) Are the generated questionnaires appropriate to be answered in a real environment (i.e., adequate number of questions)?, ($AQ3$) Is the business performance improved by using the proposed method?, and ($AQ4$) Is the proposed method preventing replanning (i.e., changing the variant which is being executed)?

Design: Two different designs are carried out in this study:

1. An *embedded* design considering Steps 2 and 3 of our approach (i.e., creating questionnaires) for addressing $AQ1$ and $AQ2$. For this, such steps are applied over a set of configurable nodes of different sizes. Such configurable nodes are part of different CBP models which are generated to represent some days of work in a business. In this design, we quantified for each configurable node: (1) the number of outgoing branches (cf. OB in Table 1), (2) the minimum, and (3) maximum number of questions which need to be answered for resolving the questionnaire associated to such node (cf. $\#mQ$ and $\#MQ$ respectively in Table 1).

   In addition, the business manager specified that answering more than 10 questions would be inefficient and thus, $AQ2$ can be answered as true if $\#MQ$ stays under 10 independently of the size of the configurable node.
2. An *holistic* design which regards the whole approach (cf. Sect. 3) is considered for addressing $AQ3$ an $AQ4$. Specifically, the current approach is applied over different CBP models each one presenting a different complexity (i.e., different number of activities). In this design, we quantified for each CBP model: (1) The number of activities of the CBP model (cf. $\#Acts$ in Table 2), (2) the number of questions which the user actually answers for individualizing the CBP model (cf. $\#Q$ in Table 2), (3) the increment of

profit which is obtained by using the current approach versus not using it (cf. $\Delta\$$ in Table 2), and (4) the percentage of cases in which replanning is avoided by using the current approach (cf. ΔR in Table 2).

Both designs are run on a Intel(R) Xeon(R) CPU E5530, 2.40GHz, 8GB memory, running Debian 6.0.3. After the application of such designs, the stored information is analyzed to answer the research questions.

Case Selection: For this case study, a real scenario which is detailed in a previous work (i.e., a beauty salon, cf. [10]) is selected. We consider this is a good and suitable case since it fulfills the following selection criteria: (1) it has been created for *an actual business*, (2) the business performance *highly relies on run-time decisions* (i.e., the knowledge of the domain expert has a great influence on the performance), and (3) the problem is *subject to variability.*

Case Study and Data Collection Procedure: A day of work in the beauty salon was modeled through a declarative specification using the language ConDec-R which was proposed in [10]. Considering the data related to each specific day of work (i.e., resource availability, services which are booked by the clients, etc.) a CBP model was generated for each day [10,9]. In addition, the salon manager provided a set of properties in form of functions (i.e., the well-defined properties written in the business language) which can be calculated from each variant which is included in the CBP model. For a period of 30 days, the following information was logged for each day through an application installed on the business:[4]

1. The CBP model which captures the different variants. As mentioned, only the best variants are kept when generating the CBP model.
2. The variant which was selected by the salon manager before starting the execution (i.e., before the first client arrived).
3. For each event that occurs during the day (e.g., when a client arrives, an activity starts or finishes), its time-stamp is recorded by the receptionist.

On the one hand, after the period of 30 days passed, for the *embedded* design, we gathered all the configurable nodes which appeared in the CBP models which were stored. Specifically, 259 configurable nodes were obtained and the current approach was applied to generate the questionnaire associated to each node. For each node, OB, $\#mQ$ and $\#MQ$ were stored. To analyze the behavior of the method against different complexities, the 259 configurable nodes were grouped considering OB. In particular 4 groups were considered: $OB \in [2, 5)$, $OB \in [5, 8)$, $OB \in [8, 11)$ and $OB \in [11, 14)$ (cf. Table. 1).

On the other hand, for the *holistic* design, the salon manager was supported by our tool. To be more precise, each time a configurable node appears (i.e., a decision needs to be taken), a questionnaire was prompted and she answered it. At the end of each day, the $\#Q$ and the selected variant (i.e., the result of the individualization) were stored. In addition, such variant was compared with

[4] The declarative model, the data which were used, and the properties which were provided can be downloaded from `http://regula.lsi.us.es/BIS/data.zip`

Table 1. Quantified variables for the embedded design

OB	$\#mQ$	$\#MQ$
$[2, 5)$	1.2	6.3
$[5, 8)$	1.1	5.0
$[8, 11)$	1.2	5.9
$[11, 14)$	1.1	6.1

Table 2. Quantified variables for the holistic design

$\#Acts$	$\#Q$	$\Delta\$$	ΔR
$(40, 60]$	3.1	141.5	70.0
$(60, 80]$	4.1	189.1	60.0
$(80, 100]$	7.6	219.4	66.7
$(100, 120]$	8.0	239.8	75.0

the variant selected before starting the execution and $\Delta\$$ was calculated and stored for each day. Furthermore, we checked if the variant which was selected before starting the execution could withstand the events logged for that day. In case it would not, we stored if replanning was avoided by using our approach, i.e., ΔR is stored.[5] The value for ΔR is calculated as the percentage of times that our approach avoided replanning against the total number of times that replanning was required. To analyze the behavior of the method against different complexities, the 30 CBP models (each one corresponding to a day of work) are grouped considering $\#Q$. In particular 4 groups are considered: $\#Q \in [40, 60)$, $\#Q \in [60, 80)$, $\#Q \in [80, 100)$, and $\#Q \in [100, 120)$ (cf. Table. 2).

Analysis and Interpretation: In order to answer $AQ1$ and $AQ2$, Table 1 is analyzed. The values of the columns $\#mQ$ and $\#MQ$ reveal that the number of questions that need to be answered for each node seems to be independent of the number of branches (cf. OB) of the related node. In addition, no errors were observed when generating the questionnaires and, thus, $AQ1$ can be answered as true. Furthermore, $\#MQ$ is lower than 10 (i.e., the number that the salon manager specified as maximum) and, thus, $AQ2$ can be answered as true.

In order to answer $AQ3$ and $AQ4$, Table 2 is analyzed. As expected, $\#Q$ increases as $\#Acts$ increases, which indicates that more effort is required by the domain expert to individualize more complex CBP models. Even though, $\#Q$ is lower than 10 in all the cases, meaning that our approach can efficiently deal with real problems. Moreover, $\Delta\$$ increases as $\#Acts$ increases, which highlights the benefits of using the proposed approach in real cases and thus, $AQ3$ can be answered as true. Regarding the values of ΔR, we can conclude that the number of times that the salon manager needs to change her initial plan due to unexpected events (e.g., a client arrives later than expected or a resource becomes unavailable) is drastically reduced (i.e., almost 43% in most complex cases). Therefore, $AQ4$ and consequently $MQ1$ can be answered as true.

Validity Evaluation: With relation to the *construct validity*, it has to be addressed in how far the measures which have been used are appropriate to address the research questions which have been planned. Firstly, the complexity of the

[5] Note that cases in which replanning becomes necessary may exist although run-time configuration is applied. In such situations, a new CBP model is created ensuring that all the included variants cover the given situation as discussed in [1,2].

problems which are considered is controlled by the number of branches of the configurable nodes and the number of activities of the CBP models in the embedded and the holistic design respectively. Although we consider that the beauty salon is a suitable business due to its complexity, different ways of varying this complexity can be applied to mitigate this threat, e.g., by changing the properties specified by the salon manager. Moreover, the duration of the logged data (i.e., 30 days) can be a threat. To the best of our knowledge there is no metric which states how long data must be logged to obtain a meaningful log. To mitigate this threat, longer durations can be considered to get more data, and therefore, to increase the probability of finding situations where the algorithm does not perform well.

Regarding the *internal validity*, the results concerning $\#MQ$ can be biased due to that the value for the upper bound of the number of questions to be answered to configure a specific configurable node (specified by the salon manager) represents a subjective point of view. This threat is difficult to eradicate. However, different business experts can be consulted to have different view points.

Finally, the *external validity* considers in how far the obtained results could be generalized to any business. This generalization is threatened by the fact that the beauty salon was the unique scenario which was studied. Other scenarios can be considered to replicate this study in order to mitigate this threat.

5 Conclusions and Future Work

To support users working on declarative BP models while a high flexibility is maintained, we propose a method which is based on removing the worst variants from the source declarative model at design time while keeping the best ones. This way, the variants which are kept are narrowed down incrementally during run-time. For this, we suggest to build upon configurable BP models. To enable configuring such models, we additionally propose to automatically generate questionnaires, unlike previous approaches. The results over a case of study are promising. As future work we plan to (1) improve the semantics of the questions which are created since they seem too artificial in some cases, (2) analyze more in depth the different classification algorithms for creating the decision trees, and (3) conduct experiments over other real scenarios for being able to generalize our results.

References

1. Barba, I., Del Valle, C., Weber, B., Jimenez-Ramirez, A.: Automatic generation of optimized business process models from constraint-based specifications. Int. J. Cooper. Inform. Syst. 22 (2013)
2. Barba, I., Weber, B., Del Valle, C., Jimenez-Ramirez, A.: User recommendations for the optimized execution of business processes. Data & Knowledge Engineering 86, 61–84 (2013)
3. Business Process Model and Notation (BPMN), Version 2.0. (2011), `http://www.omg.org/spec/BPMN/2.0/` (accessed June 1, 2011)

4. Breiman, L.: Classification and regression trees. The Wadsworth and Brooks-Cole statistics-probability series. Chapman & Hall (1984)
5. Brereton, P., Kitchenham, B., Budgen, D.: Using a protocol template for case study planning. In: Proceedings of EASE 2008. BCS-eWiC (2008)
6. Charfi, A., Müller, H., Mezini, M.: Aspect-oriented business process modeling with AO4BPMN. In: Kühne, T., Selic, B., Gervais, M.-P., Terrier, F. (eds.) ECMFA 2010. LNCS, vol. 6138, pp. 48–61. Springer, Heidelberg (2010)
7. Ferreira, H.M., Ferreira, D.R.: An integrated life cycle for workflow management based on learning and planning. Int. J. Cooper. Inform. Syst. 15(4), 485–505 (2006)
8. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H., La Rosa, M.: Configurable workflow models. J. of Cooper. Inform. Syst. 17(2), 177–221 (2008)
9. Jimenez-Ramirez, A., Barba, I., Del Valle, C., Weber, B.: Generating multi-objective optimized configurable business process models. In: RCIS 2012, pp. 1–2 (2012)
10. Jiménez-Ramírez, A., Barba, I., del Valle, C., Weber, B.: Generating multi-objective optimized business process enactment plans. In: Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) CAiSE 2013. LNCS, vol. 7908, pp. 99–115. Springer, Heidelberg (2013)
11. Karim, A., Arif-Uz-Zaman, K.: A methodology for effective implementation of lean strategies and its performance evaluation in manufacturing organizations. Business Process Management Journal 19(1), 169–196 (2013)
12. Rosa, M.L., Dumas, M., ter Hofstede, A.H.M.: Modelling business process variability for design-time configuration. In: Handbook of Research on Business Process Modeling (2008)
13. Rosa, M., Aalst, W.P., Dumas, M., ter Hofstede, A.H.M.: Questionnaire-based variability modeling for system configuration. Software & Systems Modeling 8(2), 251–274 (2009)
14. La Rosa, M., Dumas, M., Uba, R., Dijkman, R.: Merging business process models. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6426, pp. 96–113. Springer, Heidelberg (2010)
15. Rosa, M.L., Dumas, M., Uba, R., Dijkman, R.M.: Business process model merging: An approach to business process consolidation. ACM Transactions on Software Engineering and Methodology (TOSEM) (2012)
16. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. Inform. Syst. 32, 1–23 (2007)