A Method for Modeling Process Performance Indicators Variability Integrated to Customizable Processes Models

Diego Diaz[1(✉)], Mario Cortes-Cornax[1], Agnès Front[1], Cyril Labbe[1], and David Faure[1,2]

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
{diego.diaz,mario.cortes-cornax,agnes.front,
cyril.labbe}@univ-grenoble-alpes.fr
[2] Groupe INCOM & COSI+, Grenoble, France
dfaure@incom-sa.fr

Abstract. Process Performance Indicators (PPIs) are quantifiable metrics to evaluate the business process performance providing essential information for decision-making as regards to efficiency and effectiveness. Nowadays, customizable process models and PPIs are usually modeled separately, especially when dealing with PPIs variability. Likewise, modeling PPI variants with no explicit link with the related customizable process generates redundant models, making adjustment and maintenance difficult. The use of appropriate methods and tools is needed to enable the integration and support of PPIs variability in customizable process models. In this paper, we propose a method based on the Process Performance Indicator Calculation Tree (PPICT), which allows to model the PPIs variability linked to customizable processes modeled on the Business Process Feature Model (BPFM) approach. The Process Performance Indicator Calculation (PPIC) method supports PPIs variability modeling through five design stages, which concerns the PPICT design, the integration of PPICT-BMFM and the configuration of required PPIs aligned with process activities. The PPIC method is supported by a metamodel and a graphical notation. This method has been implemented in a prototype using the ADOxx platform. A partial user-centered evaluation of the PPICT use was carried out in a real utility distribution case to model PPIs variability linked to a customizable process model.

Keywords: Process performance indicators · Process families · Variability

1 Introduction

Models that support variability and customization of Business Processes (BP), i.e., process variants, have been widely studied [1, 2]. However, the variability of Process Performance Indicators (PPIs) has not been addressed in the same way [3, 4]. PPIs are quantifiable metrics that allow an evaluation of the efficiency and effectiveness of business processes. PPIs can be measured directly by generating data through the process

S. Cherfi et al. (Eds.): RCIS 2021, LNBIP 415, pp. 72–87, 2021.
https://doi.org/10.1007/978-3-030-75018-3_5

flow [5]. Decision-makers identify PPIs to get the necessary information to compare current process performances with a required objective and thus determine fundamental actions to reach proposed goals [6]. In the context of customizable processes, organizations adapt their business processes and thus their PPIs according to customers' requirements, policies or audit entities evaluations criteria. The business process variability makes PPIs definition and calculation difficult since processes and PPIs are tightly related. The performance evaluation of business processes is focused on the definition of performance requirements, e.g., a set of PPIs [7]. The design of PPIs is a time-consuming and error prone task, highly dependent on the expert know-how, which makes it difficult to integrate the modeling of customizable processes [5]. PPIs management, in the context of customizable processes, is not only delimited to the evaluation phase of business process variants, but also includes PPIs redefinition that must be carried out throughout the whole lifecycle of BP [8]. Therefore, a method that helps and promotes PPIs modeling and reusability is necessary to evaluate the performance of customizable process models.

Works related to Business Process Model Families (BPMFs) [9] and the identification of the variability of PPIs [8] respond in part to our need. For instance, the Business Process Feature Model (BPFM) is an approach to model process families, i.e., customizable processes. BPFM extends the Feature Models, which is a classic representation of software product lines variability [10, 11]. Customizable process models capture a family of process model variants in a way that the individual variants can be derived via transformations, e.g., adding, or deleting fragments [1]. However, customizable process models such as BPFM do not support PPIs variability. Likewise, the approaches modeling PPI variability such as PPINOT [5] are not integrated with customizable processes, as they treat PPIs variability in the context of a predefined process model.

In previous work, we presented the Process Performance Indicator Calculation Tree (PPICT) [12] in order to model the PPIs variability linked to customizable process models following some construction rules. The integration with the customizable process models, using the BPFM approach, was not formalized nor included in an overall method supported by a tool. Relying on our experience in a real industrial case, we propose the Process Performance Indicator Calculation (PPIC) method, which is based on the PPICT to integrate PPIs variability to customizable processes models. The contribution in this paper is threefold: I) a method of five design stages to facilitate the design and use of the PPICT, II) a metamodel to formalize the PPICT and its corresponding graphical notation, and, III) a prototype supporting the method. The method is illustrated in the context of public services distributors. Software publishers such as INCOM[1], provide these processes and PPIs to distributors. Processes that are evaluated differently by public services stakeholders as decision-makers and utility regulatory entities [13].

The rest of the paper is structured as follows. Section 2 presents in detail problems related to the modeling of PPIs variability with customizable process models. Section 3 presents our method to integrate PPIs variability modeling within BPFM. Section 4 formalizes the PPICT through a metamodel. The PPIC method validation though a user-centered evaluation is shown in Sect. 5. Section 6 discusses related works. Finally, Sect. 7 concludes, summarizes, and presents the considered perspectives.

[1] www.incom-sa.fr.

2 Modeling PPIs Variability in Customizable Process Models

Our first objective in this paper is to formalize and tool up the PPICT, which allows to integrate the PPIs variability modeling with customizable process models. The PPIs variability modeling is also called PPIs families modeling [12]. The PPICT of the Fig. 1 (b), models a family of PPIs used to evaluate a reference process dealing with the creation of contracts for utility distributors. This family of PPIs has been developed and validated in collaboration with PPIs expert developers of INCOM, a French software publisher that works for 250 public services distributors. Every distributor evaluates its own processes under different criteria, in part because these processes are variants of INCOM's reference processes. Indeed, customers can customize their software solution depending on their needs. The customizable process model Create Contract, Fig. 1 (a) is modeled using the aforementioned Business Process Feature Model (BPFM), which provides a global representation of all process variants [14]. Using a similar approach, the PPIs variability integrated to customizable process models, is represented using the so-called Process Performance Indicator Calculation Tree (PPICT) [12]. The PPICT provides a global representation of the PPIs variability definitions and calculations in a given domain through the systematic modeling of variability and common points. The PPICT defines the available PPIs members of a family of PPIs, as well as dependencies between them. A PPI in the PPICT corresponds to a query that results in an aggregation of several tuples. The "query view" of the PPICT is explained in [12] and illustrated in the next section.

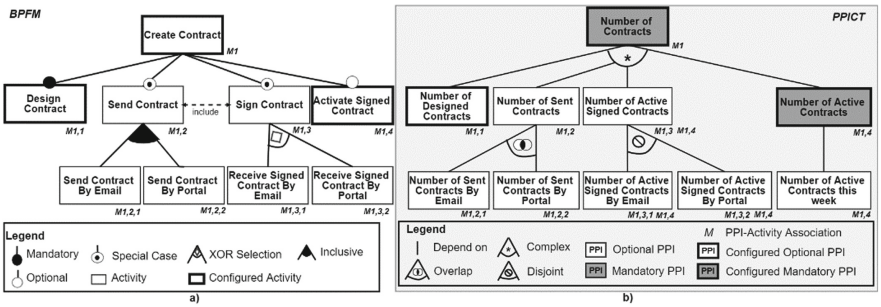

Fig. 1. (a) Customizable process model using BPFM, (b) PPICT based on the family *Number of Contracts*

To model the PPIs variability integrated to customizable process models, we rely on the graphical notation described in Sect. 4. The PPICT organizes a set of PPIs as a tree, where the tree's root identifies a PPIs family, cf. Fig. 1 (b). Each PPI of the internal tree's structure is a *Reference PPI*, i.e., each PPI that is not a tree leaf is a reference PPI including the root. Regarding PPIs-leaf, they are variants of a higher-level PPI, called *Variant PPI*. Thus, all PPIs of the internal structure except for the PPI-root are also variants of a higher-level PPI, i.e., the only PPIs that have a single role are the PPIs-leaf with variant role and the PPI-root with the reference role, cf. Fig. 1 (b). A reference PPI is a PPI that serves as the basis for calculating its variant PPIs, e.g.,

Figure 1 (b) shows that the Number of Contracts is the reference PPI of the Number of Actives Contracts. Additionally, the resulting tuples of a reference PPI contain all resulting tuples of its variant PPIs, i.e., all resulting tuples of variant PPIs are subsets of resulting tuples of its reference PPI. A variant PPI is a PPI derived from its reference PPI. This means that a variant PPI has only one reference PPI that meets this condition. Moreover, the PPICT allows to include some PPIs definitions such as, Optional PPI, Mandatory PPI, Configured Optional PPI and Configured Mandatory PPI. Likewise, the PPICT allows to integrate connections between reference PPIs and variant PPIs, which are called Overload Constraint, Disjoint Constraint, Complex Constraint and Depend on Constraint.

To integrate the PPIs variability model to the BPFM, PPICT proposes the PPI-Activity association (M), which defines that an activity can have zero or several associated PPIs and that a PPI must have at least one associated activity to be calculated: for example, the Number of Contracts PPI, Fig. 1 (b) is linked to the Create Contract activity, Fig. 1 (a). In this paper, we propose a five-stage method to facilitate the construction and use of the PPICT formalizing the PPIs variability modeling linked to customizable process models supported by a metamodel and a graphical notation.

3 PPIC Method

This section presents the Process Performance Indicator Calculation (PPIC) method, which has been partially implemented in a prototype developed on ADOxx platform. This platform allows to guide users in the development and instantiation of a metamodel. The PPIC method extends the BPFM method [14] by integrating design stages to model and calculate PPIs within customizable process models. The PPIC method is divided in 5 steps: I) the construction of the PPICT, II) the PPIs design, III) the BPFM and PPICT association, IV) the configuration of PPIs, and, V) the configuration checking concerning the business process variant. These 5 steps are described below and illustrated in Fig. 2. An example relying on a simplified INCOM's industrial case is systematically given for every step. The example is based on the Create Contract Process presented in the previous section.

Step 1, PPICT Design: refers to the manual addition of all PPIs family members using the PPICT graphical notation. PPICT Design allows to represent PPIs variability by adding PPIs depending on stakeholders' requirements. This stage must be carried out by a competence center, which includes domain experts, BP designers and decision makers to build the PPIs family.

Example: in this step, we design the PPICT in the following ways: I) by adding the PPI root into the PPIs family tree to evaluate a BP family, or, II) by adding new PPIs variants of existing PPIs according to stakeholders' requirements. In our example, the PPIs have the form Number of, as decided by the competence center. A PPICT like the one illustrated in Fig. 3 is the output of this stage and which has been implemented in our prototype.

Step 2, PPIs Design: specifies that all PPIs family members must be designed according to stakeholders' definitions and design criteria as detailed in the *PPI class* in Sect. 4. This stage must be carried out by a competence center, which includes experts in domains,

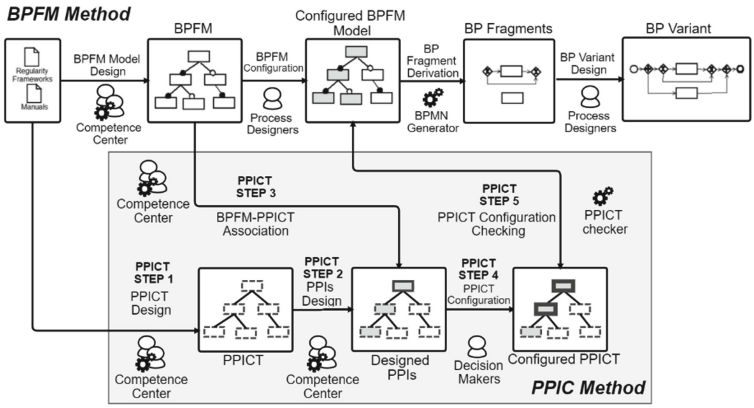

Fig. 2. PPIC method's steps

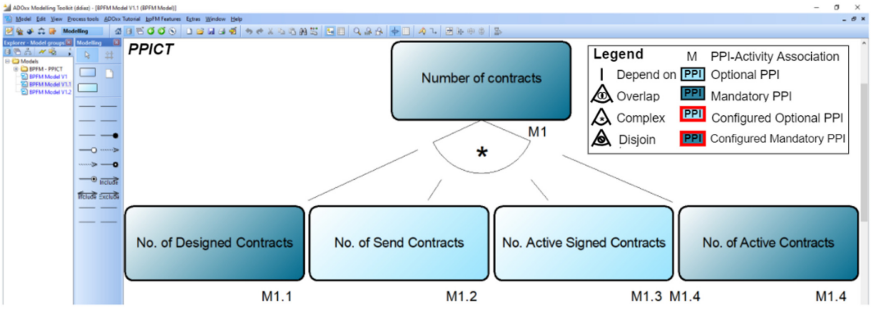

Fig. 3. PPICT (numbers of contracts PPIs family model)

BP designers and PPIs calculation experts to implement all PPIs family members as a query, e.g., as a SQL query.

Example: at this stage, PPIs are designed according to PPI class attributes detailed in Sect. 4. For instance, the Root reference PPI Number of Contracts is a mandatory PPI with a measure type Number, a measure aggregation count and a measure representation Value. A PPICT like the one illustrated in Fig. 4 (b) using SQL queries, is the output of this stage.

Step 3, BPFM-PPICT Association: allows the BPFM and PPICT association. This stage uses the PPI-Activity Mapping constraint of the PPICT relying on BPFM Model to associate reference PPIs and variant PPIs to process activities. This stage must be done manually using PPICT's PPI-Activity Mapping constraint described in [12].

Example: Fig. 4 (a) shows all activities of the Create Contract family, which are described using BPFM notation [14]. Figure 4 (b) shows all PPIs designed to evaluate the Create Contract process Family. The mappings PPI-Activity are shown here. They are done relying on each activity or group of activities that are linked to the data model of a software application, i.e., activities that generate data during their execution. For this, we integrate the process flow execution record present by default in some software

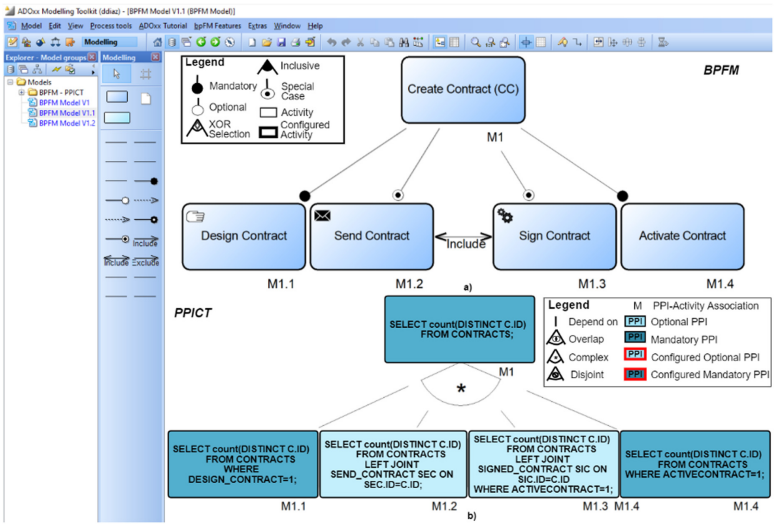

Fig. 4. Mapping PPI-activity: (a) BPFM; (b) PPICT using SQL queries

applications such as the INCOM's one. This record contains all tables and attributes used by each activity.

Step 4, PPICT Configuration: defines the PPICT configuration, i.e., the PPIs family members configured by the client. The configuration of a PPI depends on I) mandatory indicators required by regulatory entities, and, II) stakeholders' specifications to evaluate their BP variants. PPICT Configuration allows to define which PPIs family members must be included into the BP variant considering on business regulations and decision makers criteria. This step is done semi-automatically, since mandatory PPIs are automatically configured, unlike optional PPIs that must be configured manually.

Example: at this stage, decision makers configure PPIs that they believe are convenient to evaluate their process variants according to stakeholders' definitions and criteria, cf. Fig. 5 (b). Decision makers can choose any optional PPI. It does not mean these PPIs are going to be deployed, since we must check PPI-Activity match using the BP variant that has been configured.

Step 5, PPICT Configuration Checking: aligns the PPICT configuration with the BPFM configuration, i.e., check if PPICT configuration matches with BPFM configuration. PPICT Configuration Checking allows to check which members of the configured PPIs family do not match with the BP configuration. Thus, the competence center must change configured PPIs to include them into BP variant or change the BP configuration. This alignment check can be done automatically. If there is any misalignment between the PPICT and the BPFM configurations, it is necessary to return to the previous steps.

Example: at this stage, the match between PPICT and BPFM model configurations is checked. If any configured PPI is mapped to any unconfigured activity, the competence center must reconfigure BPFM or PPICT to align configured members of each family. After the reconfiguration Fig. 5 (a) and (b) shows a correct alignment BPFM-PPICT implemented in our prototype.

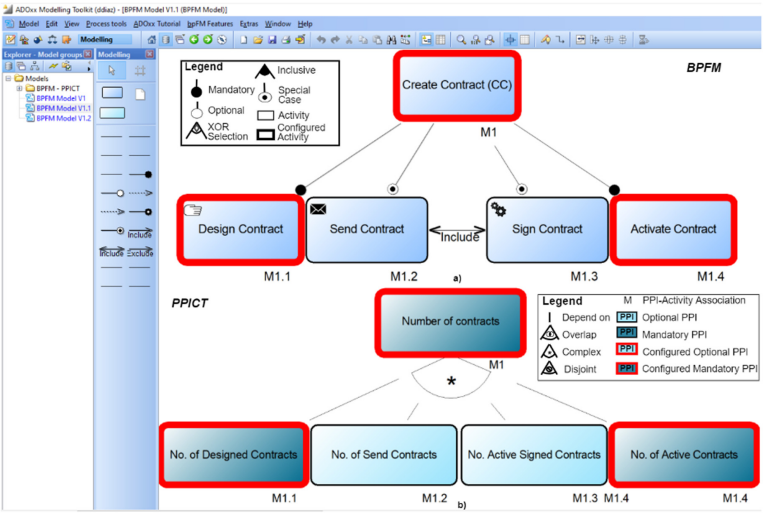

Fig. 5. (a) BPFM configuration, (b) PPICT conforming configuration

Analysts could stay in a design phase without reaching the process deployment and execution, i.e., until step 5. In this case, PPICT would be useful to analyze the feasibility of PPIs studying their alignment with the process activities in the BP family.

4 The PPICT Metamodel

This section discusses the PPICT metamodel, which extends the BPFM metamodel within the PPIs variability modeling and supports the five steps presented in the previous section. This extension is carried out according to the BP configuration and business criteria defined by stakeholders. We divide the PPICT metamodel into the following classes:

PPI class: allows to model and define all PPIs of the PPICT providing necessary information to calculate them. Every PPI must be easily identified by stakeholders, for this a short name attribute and a long name attribute are included in this class, e.g., as long name we can have *Number of Active Signed Contracts* and as short name NASC. Moreover, the selection of a PPI may imply the inclusion or exclusion of another PPI according to evaluation rules established by stakeholders. Additionally, all PPIs have a *Measure Type* that determines how they can be calculated depending on the result that the client is looking for. We define each measure type as follows:

– *Number* specifies the PPI calculation according to the number of tuples that validate a predicate, e.g., 83 Contracts are Actives.
– *Percentage* specifies the PPI calculation according to the percentage of tuples that validate a predicate, e.g., 60% of Contracts are Actives.
– *Proportion* specifies the PPI calculation according to the proportion between tuples and a target value, e.g., (3/5) 3 out of 5 Contracts are Actives.

– *Delay* specifies the PPI calculation according to the difference between creation dates of tuples, e.g., 3 Delay Days in Activating Contracts ($Date_{today} - Date_{deadline}$).
– *Respect Rate* specifies the PPI calculation according to the proportion of the difference between two dates and a target value, e.g., (3/2), 3 Current Delay Days in Activating Contracts compared to 2 Maximum Delay Days Allowed by Law ($Date_{today} - Date_{deadline}$)*/Value.*

Furthermore, every PPI has a *Measure Representation* to visualize resulting tuples in different ways depending on the type of information that the decision-maker wants to analyze, cf. Fig. 6. We define each measure representation as follows:

– *Value* representation is a type of representation that allows to visualize a set of resulting tuples as a value, e.g, 83, 60%, 5/3, 3 or 3/2.
– *Listing* representation is a type of representation that allows to visualize a set of resulting tuples as a listing. It requires to include additional projections to analyze complementary information linked to resulting tuples, e.g., *Contract Creation Date, Contract Activation Date, Contract holder, Holder's phone, among others.*
– *Geographical* representation is a type of representation that allows to visualize a set of resulting tuples geographically. It requires to group the geographical tuples, e.g., *by City, by Type of Contract, among others.*
– *Chart* representation is a type of representation that allows to visualize a set of resulting tuples as a Chart. It requires to group tuples regardless of their type, e.g., *by Year, by Type of public service, among others.*

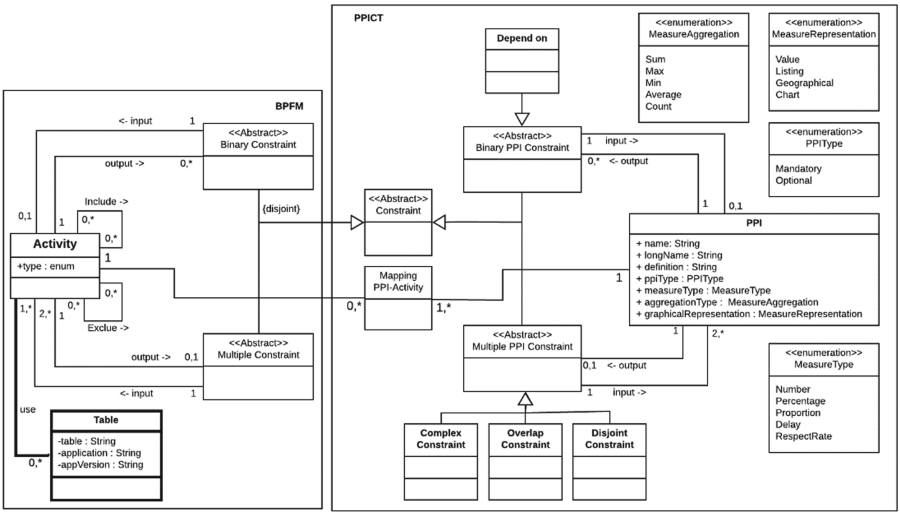

Fig. 6. PPICT metamodel linked to BPFM Metamodel

Likewise, every PPI has the attribute *Measure Aggregation* to aggregate resulting tuples in different ways, e.g., *Sum, Count, Avg, Min* and *Max*. It depends on the type of performance indicator that the decision maker wants to analyze, cf. Fig. 6.

Constraints Class: the PPICT constraints are divided into 3 groups [12]: binary constraint, multiple constraint, and PPI-activity mapping constraint. These Constraints represent (I) dependencies between reference PPIs and variant PPIs, and, (II) associations between PPIs and activities. According to [12] a PPI can be the reference for several individual variants, but a variant has only one reference.

Concerning the binary relation between PPIs, each individual variant PPI added to the tree must be connected to its reference PPI using a binary constraint. Below the definition and an example of the PPICT binary constraint:

– A *Depend-on* Constraint specifies that all resulting tuples of the connected variant PPIs are a subset or equal to resulting tuples of its reference PPI (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (a). for concrete syntax). For example, a contract that was activated this week will be part of the PPI *Number of Active Contracts this week,* but it will be also a part of the PPI reference *Number of Active Contracts* cf. Fig. 1 (b).

Every group of variant PPIs added to the tree must be connected to its reference PPI as multiple relationship. We use the PPICT Multiple Constraints presented in previous works [12], which specify the dependency between a reference PPI and a group of variants PPIs. Nevertheless, these Multiple Constraints were not considered a *Complex Constraint*, which we can describe as a combination between the existing constraints, *Overlap* and *Disjoint* proposed by [12]. That is why, we include the complex constraint to the PPICT in this paper. We know that a PPI can be the reference of a group of variants, but each variant must have only one reference. Below the definitions and examples of PPICT Multiple Constraints are detailed:

– An *Overlap Constraint* specifies that all intersections between variant PPIs are a subset or equal to resulting tuples of its reference PPI, i.e., all intersections between variant PPIs are overlap sets. (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (b) for concrete syntax). An example of this constraint can be when a contract is sent by email **and** by a web portal cf. Fig. 1 (a), this contract will be part of the PPI *Number of Send Contracts by Email* **and** part of the PPI *Number of Send Contracts by Portal,* cf. Fig. 1 (b).
– A *Disjoint Constraint* specifies that all intersections between variant PPIs are equal to zero ∅, i.e., all intersections between variant PPIs are disjoint sets (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (c). for concrete syntax). An example of this constraint can be when a user should sign contract, it could be done either by email **or** by a web portal but not by both platforms, cf. Fig. 1 (a). Hence, the signed contract will be part of the PPI *Number of Send Contracts by Email* **or** part of the PPI *Number of Send Contracts by Portal,* cf. Fig. 1 (b).
– A *Complex Constraint* specifies that all intersections between variant PPIs can be equal to zero ∅ or not, i.e., intersections between variant PPIs can be disjoint sets or not (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (d) for concrete syntax). For instance, a contract can be designed and sent but waiting to be activated, cf. Fig. 1 (a). Hence,

this contract will be part of the PPIs *Number of Designed Contracts* and *Number of Send Contracts* but will not be part of the PPI *Number of Active Contracts,* cf. Fig. 1 (b).

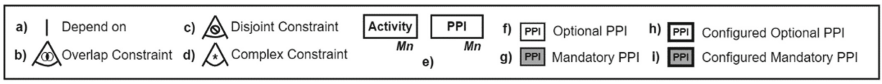

Fig. 7. PPICT concrete syntax

Additionally, a PPI can be configured optionally or not, according to the business context or decision makers requirements. Below the PPI Type is defined:

- An *Optional PPI* specifies that the PPI can be optionally configured. However, if the PPI is configured all its ascending PPIs will also be configured. (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (f) for concrete syntax).
- A *Mandatory PPI* specifies that the PPI must be configured as well as all its ascending PPIs. (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (g) for concrete syntax).
- A *Configured Optional PPI* specifies that an *Optional PPI* has been selected to be deploy for a given process variant. (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (h) for concrete syntax).
- A *Configured Mandatory PPI* specifies that a *Mandatory PPI* has been selected to be deploy for a given process variant. (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (i) for concrete syntax).

Mapping PPI-Activity Class: since PPICT is an extension of the BPFM approach, it is necessary to link the PPIs families modeling and the BP families modeling. For this, we propose the PPI-Activity association, which defines that an activity can have zero or several associated PPIs and that a PPI must have at least one associated activity to be calculated, (cf. Fig. 6 for abstract syntax and cf. Fig. 7 (e) for concrete syntax).

Table Class: all PPIs must be calculated under a specific context, according to the tables used by each activity. For this, we propose to enrich the BPFM metamodel, cf. Fig. 6 with a class that has the list of tables used by each activity. This relation Activity-Tables is usually registered in the process flow execution record presents by default in some software applications. Since every single activity may generate essential data for the PPIs calculation. The PPICT metamodel assume that the relation Activity-Tables is known and can be used to evaluate the process performance.

Note that we present here a simplified version of our PPICT metamodel, where options such as target, threshold, and worst PPIs values, as well as time conditions or state conditions are not modeled.

5 PPIC Method Validation Though a User-Centered Evaluation

An experimental protocol[2] was built relying on the THEDRE method and its decision tree MATUI [15], since this method allows to lead researches in human-centered computer science and guide researchers. We have selected this decision tree MATUI to guarantee the traceability of experimental works that were carried out. Moreover, we have divided this experimentation into two groups: experts and novices. In this analysis, we will only present the results of our experimentation with three experts since the remaining experiments had to be reprogrammed due to the current COVID-19 health crisis. The experiment description about chosen tools and produced data was essential to develop a relevant experimental protocol. Therefore, sharing a common vision between internal and external actors, i.e., researches and experts, was crucial to build the experimental material.

We have set the experimental protocol targets to achieve concerning developments, experiments, and communications, e.g., (I) involve indicator developers in the modeling of PPIs according to customizable processes, (II) explore how users express their PPIs definitions, and (III) identify users' modeling methodologies and PPIs calculation practices. Likewise, we have fixed the following Hypotheses (H) to evaluate during the experimental protocol execution:

- H1: Experts do not have a formalized method to calculate the PPIs.
- H2: PPIs are impacted by the process variability and cause uncertainty on the PPIs calculation.
- H3: Experts differentiate the main concepts of the PPCIT.
- H4: Experts take ownership of the PPCIT and understand the relationship and constraints between PPIs.
- H5: PPICT allows experts to place new PPIs in the tree structure.
- H6: Experts fill all PPI's attributes.
- H7: Existing PPIs allow experts to create new ones.
- H8: Understanding the data structure owing to the PPICT.

Regarding the experimental protocol results, experts were able to validate the PPIC method according to the PPIs variability modeling language proposed. During experiments, each hypothesis proposed some exercises based on a PPIs family scenario supported by interview guides and workbooks. Thus, experts expressed their points of view individually on either improvements or questions. Below, a synthesis about the eight hypotheses is presented:

- H1 validation: Experts do not have formal methods for calculating PPIs, but they do have practices used independently based on their experiences. Two experts apply software development methods to PPIs calculation.
- H2 validation: All experts said that *"processes variability leads to uncertainty over PPIs"* and they agree that there is a lack of reliability in PPIs calculation because they do not have tools and methods for modeling PPIs variability, e.g., tools allowing

[2] https://drive.google.com/drive/folders/1fvjgJsUu3uzbte8DL_Q5eehP4nyICiHS?usp=sharing.

users to model and calculate relationships between PPIs. Experts define the impact of process variability in different ways applying their knowledge of the trade. They said that *"the standardization of a software product is very complex"*.

- H3 validation: Experts said that *"it is indeed possible to divide the definitions of an invariable PPI and a variable PPI"*. An expert prefers simpler definitions like "*Father PPI"* and *"Son PPI"* as well as "*registrations"* instead of tuples.
- H4 validation: Several possible improvements were mentioned by the experts. For example, one of them proposed the possibility to guide user to select the correct reference PPI when new variant PPIs are added. This expert also said that *"this automatic guide to add new variant PPI can be possible through a search system allowing the indexing of the tables and fields used for each PPI"*. This improvement is very relevant for a PPICT having many modeled PPIs, that is why it can be implemented in future versions of the PPICT.
- H5 validation: Experts appreciated the notation and considered it understandable. However, an expert said that *"certain concepts need to be improved such as the constraint depends on by adding an arrow to indicate the direction of the variant*". Another expert said that *"it was difficult to place new PPIs when the labels were too long"*. The expert proposes to suggest the possible reference PPI of a new variant PPI using the labels through a search function.
- H6 validation: An expert said that *"it is much easier to add certain missing attributes to the PPIs located at the bottom of the PPICT because in general these PPIs have more information than those of the higher level"*.
- H7 validation: All experts concluded that "*the more information we have in the PPICT, the easier it is to add new PPI"*.
- H8 validation: Two experts said that "*even if some PPIs were more complicated to calculate than others, the PPICT is a tool easy to use to build PPIs and it can be used by beginners or experts concerning PPI calculation*".

These experimental results and research targets were checked such as methodological hypotheses and clarity of concepts to determine the experiments to come and PPIC method improvements. The scientific knowledge attained helped us to define the limits and advantages of our contribution, e.g., a limit is the impossibility of suggesting automatically existing reference PPIs for new variant PPIs. The prototype allowed a successful validation of the PPICT formalization facilitating the PPIs variability modeling linked to a customizable process model of our utility distributors study case.

6 Related Works

Process performance indicators are usually used to analyze the performance of business processes [7]. However, the application of PPIs in customizable process models complicates the PPIs variability modeling and management [12]. Hence, it is necessary to define new mechanisms to help PPIs developers to identify and organize the essential information to model variables performance indicator in customizable process models. Current software publishers' needs have been motivated by the measurement of

customizable process models performances. The association between process variability and performance indicators variability implies that PPIs of a non-variable process, should be redefined [8].

Regarding the performance measurement of non-customizable process models, research efforts have carried out many approaches that propose languages and architectures for monitoring and defining PPIs such as [16] or [17]. However, these approaches do not consider neither customizable process models nor the PPIs variability. Others works such as [18] have extended the Business Process Model Notation to define business process goals and performance measures, but without considering any type of variability. Regarding expressiveness of PPIs modeling, [18] has proposed a graphical notation for Business Activity Monitoring without including PPIs definitions related to data. Moreover, [19] propose an execution measurement model for business processes based on an existing software measurement ontology. But they do not consider the definition of domain-specific and user-defined PPIs. Likewise, it is also worth mentioning the standard Case Management Model and Notation for decision modelling [20], which considers the process-related measures calculation but only for non-customizable process models.

PPIs are generally defined in an informal way, e.g., in natural language, what leads to problems of ambiguity, coherence and traceability in relation to process models [21], e.g., missing information in the definition of a PPI. Likewise, PPIs are usually defined from a process variant or instance losing the perspective of the customizable process model. This entails that if a process variant evolves, PPIs definitions in the customizable process model will not be updated accordingly. On one side, the deployment of a performance management solution takes time and resources, what limits the PPIs evolutions and increases the cost for organizations [5]. On the other side, the significant gap that exists between PPIs implementation languages and natural language may cause errors in PPIs evolutions. Additionally, PPIs developers must detect and remove manually the ambiguities generated by natural language in order to calculate PPIs properly [22]. This is an error-prone task since PPIs developers often do not share the same PPIs definition as decision-makers, due to the nature of their jobs, because developers are closer to technology, while decision-makers are closer to management [5].

Customizable processes models managing either the variability by restriction or by extension have not integrated the PPIs variability modeling [1, 8]. For example, models managing variability by restriction, also called configurable process models [23], contain every possible behavior from all process variants. Thus, during the process customization the model's behavior is restricted by skipped or blocked some activities. Moreover, models managing variability by extension do not contain all possible behavior from all process variants. Instead, it represents the most common behavior in process variants and during the customization the model's behavior is extended for a specific context, e.g., new activities may be inserted to create a dedicated variant [1].

To model the PPIs variability linked to customizable processes models. We rely on the Business Process Feature Model (BPFM) [14], which includes the refinement of process variants even if the process has been customized as analyzed on [12]. This approach also considers the deployment context information adapting execution paths for every process variant. Moreover, BPFM is implemented in the ADOxx platform[3], whereby this

[3] www.adoxx.org.

approach enables to guide users to make customization decisions and prevent behavioral anomalies for every process variant. Additionally, this approach has been validated considering several Public Administration scenarios through the European Project Budget Report case study endorsed by the Learn PAd Project[4].

Classical architectures such as Data Warehouse, Business Intelligence, Business Activity Monitoring [18] or Modeling Performance Indicators [16] allow to model and calculate indicators dealing with the importance of enforcing objectives defined by business strategies and metrics. Nevertheless, in the case of customizable process models, the information extraction from business data is insufficient, especially when different PPI definitions depend on flexible evaluation criteria and process variants. The PPIs variability allows an advanced definition of variable performance indicator independently of the language used to model the BP [8]. The PPINOT approach [5] proposes a language for defining and modeling PPIs together with business processes. It enhances the PPIs modeling as well as the visual representation of business process-PPI links through a metamodel. PPINOT allows also to express PPIs definitions, which were impossible to model in previous approaches as analyzed in [21]. However, the PPINOT approach does not consider neither customizable process models nor the PPIs variability. In summary, the works of [8] and [5] do not allow to model and define relations between PPIs variability and customizable process models.

Our previous work [12] proposed the Performance Indicator Calculation Tree (PPICT), which models the PPIs variability as a tree to facilitate the PPIs definitions integrated to customizable process models. PPICT relies on BPFM constraints and proposes the term family of PPIs following the same pattern of family of processes [1]. PPICT defines a family of PPIs as a paradigm for calculating PPIs using a set of processes that form a common structure, which serves as a basis for calculating derived PPIs according to process variants and PPIs definitions.

From the study of the state of the art, we conclude that when an organization explores its data sources and uses it as part of new process, there are no design stages for customizable process models-PPIs variability links. For this reason, we propose a method that formalizes the PPICT and extends BPFM method to model and facilitate the definitions of PPI variants in the context of customizable process models.

7 Conclusion and Open Issues

Nowadays, customizable process models and PPIs are usually modeled separately, especially when dealing with PPIs variability. Since the support of process variability complicates the PPIs definition and calculation. Modeling PPI variants with no explicit link with the related customizable process generates redundant models, making adjustment and maintenance difficult. In previous work, we presented the Process Performance Indicator Calculation Tree (PPICT) [12] in order to model the PPIs variability linked to customizable process models. However, the integration with customizable process models, using the BPFM approach, had not yet been formalized nor included in an overall method supported by a tool. This paper proposes the PPIC method based on the PPICT

[4] www.learnpad.eu.

to integrate PPIs variability to customizable process models. Our contribution lies in three main axes: (I) a method of five design stages to facilitate the design and use of the PPICT, (II) a metamodel to formalize the PPICT and its corresponding graphical notation, and, (III) a prototype supporting this method developed on ADOxx Platform. The PPIC method is illustrated in a real utility distributor case and has been validated by PPI calculation experts. The validation was carried out though a user-centered evaluation, which allows to use the PPICT to model a PPIs family linked to a BP family. Additionally, the PPIC method complements the related works of customizable process models by broadening the spectrum not only of the processes variants to be measured, but also of the measures themselves through the PPIs variability modeling. Today, the prototype does not allow to execute SQL queries to calculate PPIs. The latter feature is planned to be implemented rapidly, after completing the experimentations. A deepening track would be the integration with queries execution tools as business intelligence systems to implement all PPIs family members. Another interesting improvement would be the modeling of the PPIs variability in BP families that use non-relational storage systems, e.g., by extending the PPICT notation and links between PPIs and data being data-model agnostic.

References

1. La Rosa, M., Van Der Aalst, W.M.P., Dumas, M., Milani, F.P.: Business process variability modeling: a survey. ACM Comput. Surv. (CSUR) **50**, 2 (2017)
2. Milani, F., Dumas, M., Ahmed, N., Matulevičius, R.: Modelling families of business process variants: A decomposition driven method. Inf. Syst. **56**, 55–72 (2016)
3. Domínguez, E., Pérez, B., Rubio, Á.L., Zapata, M.A.: A taxonomy for key performance indicators management. Comput. Stand. Interfaces **64**, 24–40 (2019)
4. Estrada-Torres, B.: Improve performance management in flexible business processes. In: Proceedings of the 21st International Systems and Software Product Line Conference-Volume B, pp. 145–149 (2017)
5. del Río Ortega, A., Resinas, M., Durán, A., Bernárdez, B., Ruiz-Cortés, A., Toro, M.: Visual PPINOT: a graphical notation for process performance indicators. Bus. Inf. Syst. Eng. **61**, 137–161 (2019)
6. Peral, J., Maté, A., Marco, M.: Application of data mining techniques to identify relevant key performance indicators. Comput. Stand. Interfaces **54**, 76–85 (2017)
7. Estrada Torres, B., Torres, V., del Río Ortega, A., Resinas Arias de Reyna, M., Pelechano, V., Ruiz Cortés, A.: Defining PPIs for process variants based on change patterns. In: JCIS 2016: XII Jornadas de Ciencia e Ingeniería de Servicios (2016)
8. Estrada-Torres, B., del-Río-Ortega, A., Resinas, M., Ruiz-Cortés, A.: Identifying variability in process performance indicators. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNBIP, vol. 260, pp. 91–107. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45468-9_6
9. La Rosa, M., Dumas, M., Ter Hofstede, A.H.M., Mendling, J.: Configurable multi-perspective business process models. Inf. Syst. **36**, 313–340 (2011)
10. Gröner, G., et al.: Validation of families of business processes. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 551–565. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_41
11. Villota, A., Mazo, R., Salinesi, C.: The high-level variability language: an ontological approach. In: Proceedings of the 23rd International Systems and Software Product Line Conference-Volume B, pp. 162–169 (2019)

12. Diaz, D.: Integrating PPI variability in the context of customizable processes by extending the business process feature model. In: 2020 IEEE 24th International Enterprise Distributed Object Computing Workshop (EDOCW), pp. 80–85. IEEE (2020)
13. Diaz, D., Cortes-Cornax, M., Labbé, C., Faure, D.: Modélisation de la variabilité des indicateurs dans le cadre des administrations de services publics (2019)
14. Cognini, R., Corradini, F., Polini, A., Re, B.: Business process feature model: an approach to deal with variability of business processes. In: Karagiannis, D., Mayr, H., Mylopoulos, J. (eds.) Domain-Specific Conceptual Modeling, pp. 171–194. Springer, Heidelberg (2016)
15. Mandran, N., Dupuy-Chessa, S.: Supporting experimental methods in information system research. In: 2018 12th International Conference on Research Challenges in Information Science (RCIS), pp. 1–12. IEEE (2018)
16. Popova, V., Sharpanskykh, A.: Modeling organizational performance indicators. Inf. Syst. **35**, 505–527 (2010)
17. Saldivar, J., Vairetti, C., Rodríguez, C., Daniel, F., Casati, F., Alarcón, R.: Analysis and improvement of business process models using spreadsheets. Inf. Syst. **57**, 1–19 (2016)
18. Friedenstab, J.-P., Janiesch, C., Matzner, M., Muller, O.: Extending BPMN for business activity monitoring. In: 2012 45th Hawaii International Conference on System Sciences, pp. 4158–4167. IEEE (2012)
19. Delgado, A., Weber, B., Ruiz, F., de Guzmán, I.G.-R., Piattini, M.: An integrated approach based on execution measures for the continuous improvement of business processes realized by services. Inf. Softw. Technol. **56**, 134–162 (2014)
20. OMG: Case Management Model and Notation (CMMN). 1.1 (2016)
21. Del-RíO-Ortega, A., Resinas, M., Cabanillas, C., Ruiz-Cortés, A.: On the definition and design-time analysis of process performance indicators. Inf. Syst. **38**, 470–490 (2013)
22. van der Aa, H., et al.: Narrowing the business-IT gap in process performance measurement. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 543–557. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39696-5_33
23. Reichert, M., Hallerbach, A., Bauer, T.: Lifecycle management of business process variants. In: vom Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management 1 International Handbooks on Information Systems, pp. 251–278. Springer, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-642-45100-3_11