



HAL
open science

Process Performance Indicators Variability integrated to Customizable Process Models

Diego Diaz Bohorquez

► **To cite this version:**

Diego Diaz Bohorquez. Process Performance Indicators Variability integrated to Customizable Process Models. Performance [cs.PF]. Université Grenoble Alpes [2020-..], 2023. English. NNT: 2023GRALM092 . tel-04677419

HAL Id: tel-04677419

<https://theses.hal.science/tel-04677419v1>

Submitted on 26 Aug 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : MSTII - Mathématiques, Sciences et technologies de l'information, Informatique

Spécialité : Informatique

Unité de recherche : Laboratoire d'Informatique de Grenoble

**Variabilité des indicateurs de performance des processus intégrée
aux modèles de processus personnalisables**

**Process Performance Indicators Variability integrated to
Customizable Process Models**

Présentée par :

Diego DIAZ BOHORQUEZ

Direction de thèse :

Cyril LABBE

PROFESSEUR DES UNIVERSITES, UNIVERSITE GRENOBLE ALPES

Directeur de thèse

Mario CORTES CORNAX

MAITRE DE CONFERENCES, UNIVERSITE GRENOBLE ALPES

Co-encadrant de thèse

Rapporteurs :

SAID ASSAR

PROFESSEUR, IMT BUSINESS SCHOOL

REBECCA DENECKERE

MAITRESSE DE CONFERENCES HDR, UNIVERSITE PARIS 1 - PANTHEON SORBONNE

Thèse soutenue publiquement le **20 décembre 2023**, devant le jury composé de :

GWEN SALAUN,

PROFESSEUR DES UNIVERSITES, UNIVERSITE GRENOBLE
ALPES

Président

CYRIL LABBE,

PROFESSEUR DES UNIVERSITES, UNIVERSITE GRENOBLE
ALPES

Directeur de thèse

SAID ASSAR,

PROFESSEUR, IMT BUSINESS SCHOOL

Rapporteur

REBECCA DENECKERE,

MAITRESSE DE CONFERENCES HDR, UNIVERSITE PARIS 1 -
PANTHEON SORBONNE

Rapporteuse

PATRICIA SERRANO-ALVARADO,

MAITRESSE DE CONFERENCES HDR, UNIVERSITE DE NANTES

Examinatrice

Invités :

AGNES FRONT

PROFESSEURE DES UNIVERSITES, UNIVERSITE GRENOBLE ALPES

MARIO CORTES-CORNAX

MAITRE DE CONFERENCES, UNIVERSITE GRENOBLE ALPES



Keywords

Customizable Process Models
Performance Indicators
Process Families
Process Models
Process Performance Indicators
Process Performance Indicators Variability
Process Variability
Business Process
Process Performance Indicators Family

Abstract

Process Performance Indicators (PPIs) are quantifiable metrics to evaluate the business process performance providing essential information for decision-makers. Today, Customizable Process Models and PPIs are usually modeled separately, especially PPI variables. Likewise, modeling PPI variants with no explicit link with the related Customizable Processes generates redundant models, making adjustment and maintenance difficult. The use of appropriate methods and tools is needed to enable the integration and support of PPI Variability in Customizable Process Models. In this thesis, we propose the Process Performance Indicator Calculation (PPIC) method, which allows the modeling of PPI Variability linked to Customizable Processes modeled on the Business Process Feature Model (BPFM) approach. The Process PPIC method supports PPI Variability modeling through five design stages, which concern 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 complete user-centered evaluation of the use of the PPICT method as carried out in a real utility distribution case to model PPI Variability linked to a Customizable Process Model.

Table of Contents

KEYWORDS	I
ABSTRACT	II
TABLE OF CONTENTS	III
LIST OF FIGURES	V
LIST OF TABLES	VIII
LIST OF ABBREVIATIONS	IX
ACKNOWLEDGEMENTS	X
CHAPTER 1: INTRODUCTION	1
1.1 GENERAL CONTEXT OF THE STUDY: CUSTOMIZABLE PROCESSES AND PROCESS PERFORMANCE INDICATORS	1
1.2 RESEARCH PROBLEM: THE DIFFICULT ALIGNMENT BETWEEN CUSTOMIZABLE PROCESS MODELS AND PROCESS PERFORMANCE INDICATOR VARIABILITY	2
1.3 GOALS AND CONTRIBUTIONS: CONVERGING PROCESS AND PPI VARIABILITY	7
1.4 ORGANIZATION OF THE DOCUMENT	8
CHAPTER 2: MODELING PPI VARIABILITY AND CUSTOMIZABLE PROCESS: STATE OF ART	10
2.1 CUSTOMIZABLE PROCESS MODELS	11
2.1.1 <i>Customizable Process Models requirements for PPI modeling</i>	12
2.1.2 <i>Configurable integrated Event-driven Process Chains (C-iEPCs)</i>	15
2.1.3 <i>Configurable Workflows</i>	18
2.1.4 <i>Template and rules</i>	21
2.1.5 <i>Business Process Feature Model</i>	23
2.1.6 <i>Synthesis of Customizable Process Models</i>	27
2.2 PROCESS PERFORMANCE INDICATOR MODELING APPROACHES	29
2.2.1 <i>Process Performance Indicator Requirements</i>	31
2.2.2 <i>Business Performance Measurement approach</i>	32
2.2.3 <i>Business Activity Monitoring and Business Process Execution Measurement Model</i> ..	34
2.2.4 <i>Data Warehouse Architecture</i>	36
2.2.5 <i>Business Intelligence Approach</i>	37
2.2.6 <i>PPINOT</i>	39
2.2.7 <i>Synthesis of Performance Indicator modeling</i>	43
2.3 SUMMARY AND IMPLICATIONS	45
CHAPTER 3: MODELING PROCESS PERFORMANCE INDICATORS VARIABILITY INTEGRATED TO CUSTOMIZABLE PROCESS MODELS: APPROACH OVERVIEW	46
3.1 OVERVIEW OF THE THREE MAIN PROPOSITIONS	47
3.1.1 <i>THE PPICT OVERVIEW</i>	47
3.1.2 <i>OVERVIEW OF THE PPIC METHOD</i>	48
3.1.3 <i>OVERVIEW OF PPIC PROTOTYPE</i>	50
3.2 RESEARCH DESIGN RELYING ON THE THEDRE METHOD	51
3.2.1 <i>Methodology</i>	51
CHAPTER 4: PROCESS PERFORMANCE INDICATOR CALCULATION TREE (PPICT)	54
4.1 DEFINITIONS	54
4.2 PPICT CONSTRAINTS	59
4.3 PPICT-ACTIVITY CONSTRAINT	62
4.4 THE PPICT METAMODEL	62

4.4.1	<i>PPICT Metamodel Classes</i>	64
4.4.2	<i>PPI Class Attributes</i>	67
4.4.3	<i>Direct Relation Between PPIs</i>	69
4.5	PPICT METAMODEL INSTANCE.....	70
4.6	PPICT CONCLUSION.....	71
CHAPTER 5: PROCESS PERFORMANCE INDICATOR CALCULATION METHOD (PPIC METHOD)		73
5.1	PPIC METHOD STAGES.....	73
5.2	THE PPIC METAMODEL (LINK BETWEEN THE BPFM METAMODEL AND THE PPICT METAMODEL)	81
5.3	EVALUATION OF THE PPIC METAMODEL.....	84
CHAPTER 6: VALIDATION OF THE PROCESS PERFORMANCE INDICATOR CALCULATION METHOD THROUGH A USER-CENTERED EVALUATION		86
6.1	EXPERIMENTAL PROTOCOL DESIGN	86
6.2	PARTICIPANTS	89
6.3	INSTRUMENTS USED DURING EACH EXPERIMENT	90
6.4	EXPERIMENT ANALYSIS.....	94
6.5	LIMITATIONS IDENTIFIED DURING THE EXPERIMENT	98
6.5.1	<i>Limitations of concepts and functionalities</i>	98
6.5.2	<i>Limitation of the experiment saturation</i>	99
6.6	CONCLUSION OF THE USER-CENTERED EVALUATION	101
CHAPTER 7: PROCESS PERFORMANCE INDICATOR CALCULATION TOOL (PPIC TOOL).....		102
7.1	USE OF THE TOOL FOR STEP 2 - PPI DESIGN	103
7.2	USE OF THE TOOL FOR STEP 2 - PPI DESIGN	105
7.3	USE OF THE TOOL FOR STEP 3 - BPFM-PPICT ASSOCIATION	109
7.4	USE OF THE TOOL FOR STEP 4 -PPICT CONFIGURATION	115
7.5	USE OF THE TOOL FOR STEP 5 - PPICT CONFIGURATION CHECKING	116
7.6	CONCLUSION AND LIMITATIONS	121
CHAPTER 8: CONCLUSIONS, LIMITATIONS, AND PERSPECTIVES		122
8.1	CONCLUSION	122
8.2	LIMITATIONS.....	124
PPICT Limitations	124	
PPIC Method limitations	124	
PPIC tool limitations.....	125	
PPICT and PPIC Method limitation.....	125	
PPICT and PPIC tool limitations.....	125	
8.3	PERSPECTIVES	126
8.3.1	<i>Short-term perspectives</i>	126
8.3.1	<i>Medium-term perspectives</i>	126
8.3.1	<i>Long-term perspectives</i>	126
BIBLIOGRAPHY		128
APPENDICES		131

List of Figures

Figure 1 Research Context: example of two different deployed processes showing that Business Process Variability and PPI Variability are directly connected.	3
Figure 2 Process Reference: <i>Create Contract</i> modelled in BPMN.	3
Figure 3 Process variant deployed for the utility distributor A: this process is a variant of the reference process of Figure 2.....	4
Figure 4 Process variant deployed for the utility distributor B: this process is a variant of the reference process of Figure 2.....	4
Figure 5 Example of representation for the <i>Number of Active Contracts</i>	5
Figure 6 Example of representation for the <i>Number of Active Contracts Signed by Portal</i>	5
Figure 7 Example of representation for the of the <i>Number of Active Signed Contracts</i>	6
Figure 8 Example of variability by restriction compared to (a), in (b) the activity sign contract has been removed.	13
Figure 9 Example of variability by extension, in (b) activities Send Contract by Email, Send Contract by Portal, Sign Contract by Email and Sign Contract by Portal have been added compared to (a)	13
Figure 10 Example of the C-iEPC model based on the <i>Create Contract</i> Process	16
Figure 11 (a) Example of create contract process in C-YAWL with a sample customization; (b) the customized create contract process model.....	19
Figure 12 (a) The Create Contract Process Reference in <i>Template & Rules</i> . (b) Customized model of the Create Contract Process.....	22
Figure 13 BPFM Method (Cognini et al., 2016).....	24
Figure 14 A BPFM Model created via the BPFM tool.....	25
Figure 15 VISUAL PPINOT icons for PPIs, base measures, and aggregated measures (Del-Río-Ortega et al., 2019).....	40
Figure 16 VISUAL PPINOT model example	40
Figure 17 Definition of measures in PPINOT (Del-Río-Ortega et al., 2019).....	41
Figure 18 PPICT example.....	48
Figure 19 Steps of PPIC Method	50
Figure 20 Ph.D. process research based on The THEDRE Process (Mandran & Dupuy-Chessa, 2018).....	53
Figure 21 Family of PPIs: PPI Query and relationship.....	55
Figure 22 Data base model of the family of PPIs	56

Figure 23 Process Performance Indicator Calculation tree (internal tree's structure)	59
Figure 24 PPICT Constraints and Rules: each constraint respects one or more modeling rules.....	61
Figure 25 PPICT-BPFM CONSTRAINT	62
Figure 26 Metamodel of Process Performance Indicator Calculation Tree PPICT.....	63
Figure 27 PPICT Concrete Syntax.....	64
Figure 28 PPICT Metamodel instance	71
Figure 29 PPIC Method steps (bottom box) and BPFM Method (top box).....	73
Figure 30 Create Contract Process modeled using BPFM notation.....	74
Figure 31 Example of step 1: Family <i>Number of Contract</i>	76
Figure 32 Result of step 2 PPI Design – Mandatory PPI (natural language view)	77
Figure 33 Example of BPFM-PPICT Mapping (M)	78
Figure 34 Example of Activity generating data	78
Figure 35 Example of PPI design (SQL view).....	78
Figure 36 PPICT Configuration: configuration of <i>Number of contracts</i> , <i>Number of Active Contracts</i> and <i>Number of Designed Contracts</i>	79
Figure 37 Miss-mapping between BPFM and PPICT because Activate Signed Contract Activity is not configured in the BPFM, but the <i>Number of Active Contracts</i> is configured.....	80
Figure 38 Correct BPFM-PPICT mapping: the PPI <i>Number of Active Contracts</i> is aligned with the activity <i>Active Contract</i>	81
Figure 39 PPIC Metamodel: link with the BPFM metamodel and PPICT Metamodel.	83
Figure 40 Experimental steps instantiating the Experiment phase of the THEDRE method.....	87
Figure 41 MATUI decision tree (Mandran & Dupuy-Chessa, 2018), pink way in our case due to a dynamic experiment.....	89
Figure 42 Example of experimental material (Questionnaire).....	91
Figure 43 Experimental Material: workbook.....	92
Figure 44 Matrixes to evaluate the methodological aspects and proposed concepts.....	93
Figure 45 Contact time by single case interviews based on the analysis of (Marshall et al., 2013).....	101
Figure 46 ADOxx Meta Modelling Platform Hierarchy	103
Figure 47 PPICT design of the <i>Number of Contracts</i> PPI family.....	104
Figure 48 Description of PPI <i>Number of Contracts</i>	105

Figure 49 General description of the PPI *Number of Contracts* 107

Figure 50 PPI Conditions for the PPI *Number of Contracts*..... 108

Figure 51 Linear Evaluation: Desired thresholds to evaluation of the PPI results .. 108

Figure 52 *Number of Contract* PPI family designed the PPICT tool..... 109

Figure 53 Selection of the activity associated to the PPI *Number of Contracts* 110

Figure 54 Selection of the associated PPI for the *Create Contract* activity. 111

Figure 55 BPFM-PPICT Association 112

Figure 56 Query association of the PPI *Number of Contracts*..... 113

Figure 57 PPI design (SQL execution) 114

Figure 58 PPI *Number of Active Contracts* and *Number of Active Contracts By City*..... 114

Figure 59 PPI *Number of Active Contracts Signed by Portal* and *Number of Active Contracts by City* 115

Figure 60 PPI Configuration sheet on our PPIC tool..... 116

Figure 61 PPICT Configuration on our PPIC tool..... 116

Figure 62 Mismatching between BPFM (*Activate Contract*)-PPICT (*Number of Active Contracts*) a) BPFM Configuration, b) PPICT configuration. 118

Figure 63 Configuration of the *Active Contract* Activity in the BPFM model..... 119

Figure 64 Correct BPFM-PPICT mapping a) BPFM Configuration, b) PPICT configuration 120

List of Tables

Table 1 Requirements analysis of the C-iEPC approach	17
Table 2 Requirements analysis of the Configurable Workflows approach	20
Table 3 Requirements analysis of the <i>Template and rules</i> approach.....	23
Table 4 Requirements analysis of the BPFM approach	26
Table 5 Evaluation of Customizable Process Model Approaches	28
Table 6 Example of ARIS tool in the context of medical distribution	30
Table 7 Requirements analysis of the Business Performance Measurement BSC approach.....	33
Table 8 Requirements analysis of the BPEMM approach	35
Table 9 Requirements analysis of the Data Warehouse Architecture approach (Ngo et al., 2019)	36
Table 10 Requirements analysis of the Business intelligence BIA approach	38
Table 11 Requirements analysis of the PPINOT approach.....	41
Table 12 Evaluation of Process Performance Indicators Approaches	44
Table 13 Process flow execution record	82
Table 14 Professional experience of participants.....	90
Table 15 Experimental material and experimental guide that help to validate hypotheses.....	93
Table 16 Synthesis of hypothesis validation.....	97
Table 17 Limits of the PPICT and PPIC Method	98
Table 18 Interviewees, Interviews, and Contact Time by Research Design (Marshall et al., 2013).....	100

List of Abbreviations

Business Process Feature Model (BPFM)
Business Process Model Families (BPMFs)
Process Performance Indicator Calculation (PPIC)
Process Performance Indicator Calculation Tree (PPICT)
Process Performance Indicators (PPIs)
Key Performance Indicators (KPIs)
Customizable Process Models (CPMs)
Association nationale de la recherche et de la technologie (ANRT)
Object Management Group (OMG)
Business Process (BP)
Business Process Management (BPM)
Business Process Model Notation (BRMN)

Acknowledgements

I would like to express my deep and sincere gratitude to my research supervisors for their continued support and encouragement: Professor Cyril LABBE; MCF, HDR. Mario CORTES-CORNAX and Professor Agnès FRONT, my company's supervisor David FAURE and my committee chairs: Professor Said, ASSAR; MCF, HDR Rebecca, DENECKERE; Professor Gwen, SALUN; MCF, HDR Patricia, SERRANO-ALVARADO. I offer my sincere appreciation for the learning opportunities provided by my committee.

My completion of this thesis could not have been accomplished without the support of Mario CORTES-CORNAX, thank you for giving me your time and your experience in research and writing. Thanks to all the participants for having helped me with the experimental protocol. Thanks to my parents as well, Mr. Belisario DIAZ. and Ms. Rosa BOHORQUEZ for the countless times you encouraged me during our hectic schedules. Thanks to my future wife for motivating me and helping me complete my research tasks.

Chapter 1: Introduction

1.1 GENERAL CONTEXT OF THE STUDY: CUSTOMIZABLE PROCESSES AND PROCESS PERFORMANCE INDICATORS

Faced with the rapid growth of software publishers, business leaders have increasingly felt the need for adaptable decision-making. The effectiveness of decision-making depends on the provision of relevant information and appropriate tools. The problem for software publishers is to effectively adapt reports and indicators to specific clients depending on their criteria and the business processes deployed in their companies.

A business process can be defined as a set of tasks to deliver a service or product (OMG, 2016). A business process can also be defined as a set of activities to fulfill one of the business goals (OMG, 2016). Process Performance Indicators (PPIs) are used to measure business process performance. PPIs evaluate the process goals that business leaders want to achieve. Traditional decision-making systems using PPI provide single decision support without considering the possibility to customize the process model. A customizable process refers to a business process (so-called business process reference) that can be personalized into variants for a specific context depending on business criteria. A set of business process references and business process variants are also called Business Process Family and concern the domain of Business Process Variability (Schnieders & Puhlmann, 2006). Decision-making systems do not integrate the potential Process Variability in the calculation of PPIs. The so-called PPI Variability is a domain that seeks to formalize the diverse ways to design and calculate indicators to analyze Business Process Models.

Our research work has been performed at the LIG laboratory (Grenoble Computer Science Research Institute) within the team SIGMA (Information Systems - Adaptable Engineering and Modeling). The team has obtained experience in design, modeling, and development of complex systems. Our work is partially financed by the ANRT (National Association of Research and Technology) in the context of a CIFRE

PhD¹ society. INCOM is a software publisher in the field of utility distribution with more than 250 clients. The application framework of our work is in the field of utility distribution supported by INCOM's suite. INCOM exploits large volumes of operational information from their clients. An adaptation of INCOM's information system is needed, for new clients' requirements and new policies of evaluations of audit entities. The INCOM suite is constituted of its solutions: Anemone, for customer relationship management, Activ'tech for meter reading and Portail for online user services. The business processes supported by these solutions are customized according to the client's needs. Each client potentially has different indicators to analyze the deployed processes. Current commercial packages of software publishers include analysis tools to monitor common processes. These tools do not offer an integrated solution for the design and development of decision-making systems for Business Process Variability and PPIs Variability.

In the context of heterogeneous deployed Process Models, the design and development of decision-making systems has proved to be essential. By nature, public sector' software solutions require complex data structures. For instance, each public entity brings a set of varied business processes and tools such as administrative processes, intervention processes and decision-making reports. The complexity of these structures makes it difficult to consider PPIs Variability with current Customizable Process Models due to their low level of abstraction in the performance evaluation. The research problem is illustrated in the next sub-chapter through a case study of Customizable Process Models in the field of utility distribution.

1.2 RESEARCH PROBLEM: THE DIFFICULT ALIGNMENT BETWEEN CUSTOMIZABLE PROCESS MODELS AND PROCESS PERFORMANCE INDICATOR VARIABILITY

Customized Process Models in the context of utility distribution require aggregation mechanisms and individual decision support systems to synthesize the information. These mechanisms are becoming crucial since they are instrumental for the analysis of all process variants from a single decision-making system.

¹ <https://www.incom-sa.fr/>

INCOM needs to evaluate Customizable Process Models and model PPI Variability, to calculate or recalculate the PPIs that vary from one utility distributor to another. We explain the PPIs Variability through one of the client processes most used by INCOM’s clients, called *Create Contract* (cf. Figure 1). INCOM customizes and deploys this process for every utility distributor, e.g., utility distributor A and utility distributor B in Figure 1. The variants A and B correspond to a customization of the *Create Contract* process (the reference) due to the distributor’s internal organization and certain general rules adapted to their specific know-how. The *Create Contract Process* is composed of 4 main activities: design contract, send contract, sign contract and active contract. Those activities can be defined as subprocesses depending on the client’s criteria to deploy the process variant. Figure 2 shows a simplified version of the *Create Contract* reference process using the standard Business Process Model and Notation (BPMN) (OMG, 2016), which provides a graphical representation of the business process.

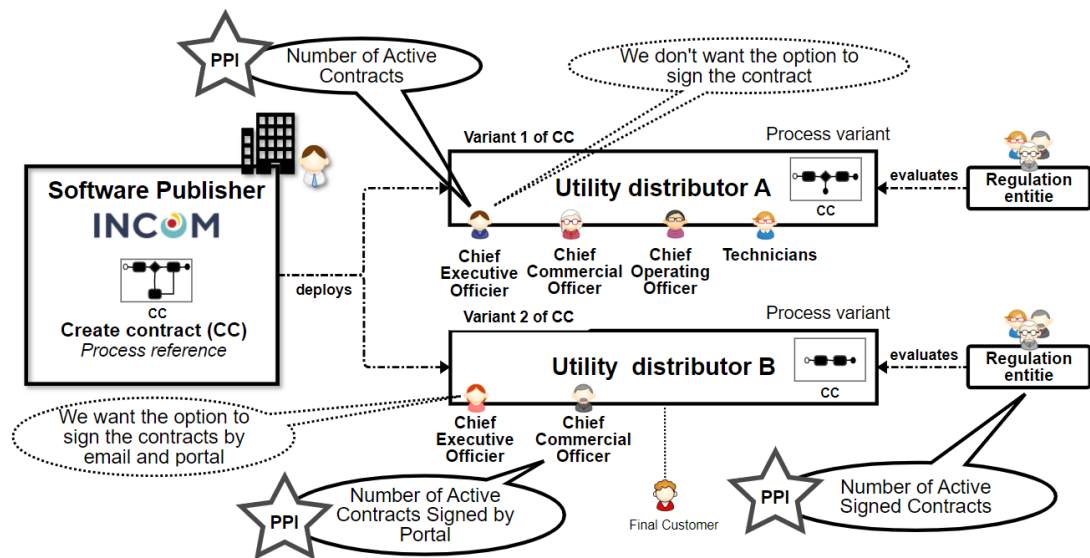


Figure 1 Research Context: example of two different deployed processes showing that Business Process Variability and PPI Variability are directly connected.

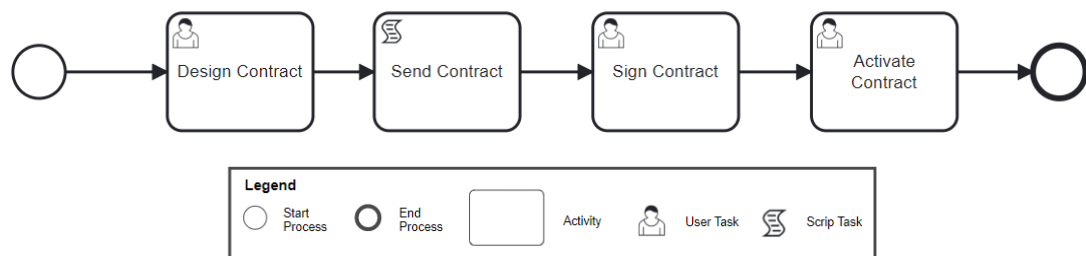


Figure 2 Process Reference: *Create Contract* modelled in BPMN.

A simple process variant of the utility distributor A is illustrated in Figure 3. Since the client has decided not to include the activity to sign the contract, the process variant only has three activities: *design contract*, *send contract* and *activate contract*.

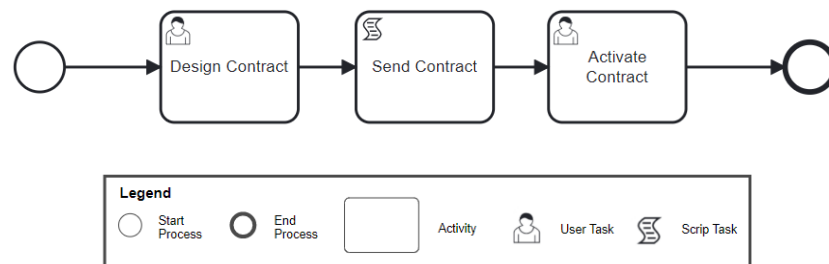


Figure 3 Process variant deployed for the utility distributor A: this process is a variant of the reference process of Figure 2

A possible process variant of the utility distributor B is illustrated in Figure 4. Since the client has decided to include the activities to sign the contract by email and by portal, INCOM must also deploy the options to send the contract by email and by portal. The activities *send contract* and *sign contract* are in this case subprocesses, since it is necessary to detail the tool used to send and sign the contract.

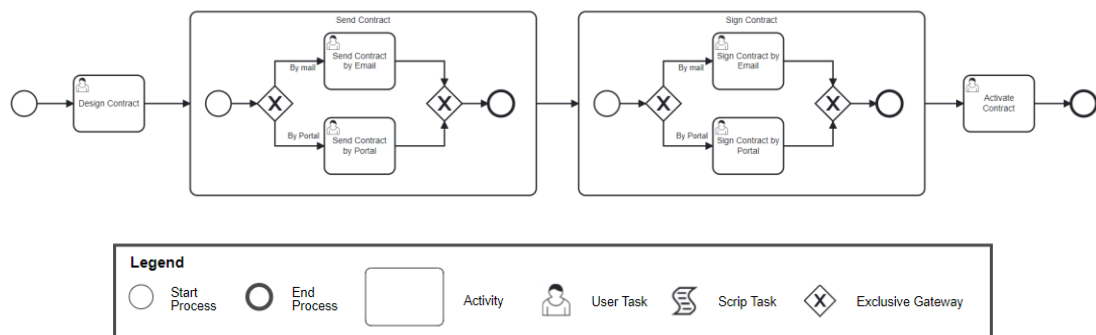


Figure 4 Process variant deployed for the utility distributor B: this process is a variant of the reference process of Figure 2

INCOM must also provide their clients with the relevant indicators regarding the stakeholders' requirements, i.e., regulation entities and employees' requirements (cf. Figure 1). For instance, the CEO of utility distributor A wants to know the *Number of Active Contracts*. Additionally, since client A does not want the option to sign the contracts, all indicators related with this activity cannot be calculated for this process variant (cf. Figure 1). Distributor B wants the option to sign the contract by email and via the portal and it's regulation entity wants to know the *Number of Active Signed Contracts*. Therefore, INCOM must deploy the activity to sign contracts. Moreover, the CEO of distributor B wants to know the *Number of Active Contracts Signed by*

Portal. This implies that INCOM must deploy the activity to manage the signature of contracts by the portal. Thus, the different definitions of a PPI and the deployed process, the calculation of this indicator will vary, e.g., for the PPI *Number of Active Contracts*, the representation could be as presented in Figure 5.

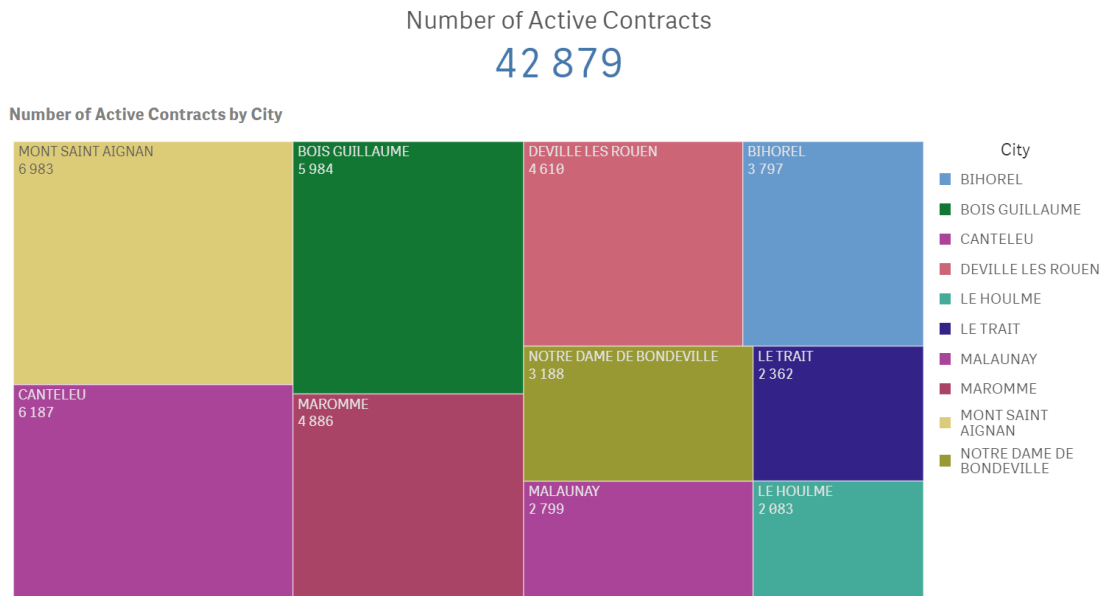


Figure 5 Example of representation for the *Number of Active Contracts*

Regarding the PPI *Number of Active Contracts Signed*, the representation could be as presented in Figure 6.



Figure 6 Example of representation for the *Number of Active Contracts Signed by Portal*

Regarding the PPI *Number of Active Signed Contracts by portal*, the representation could be presented as in Figure 7:

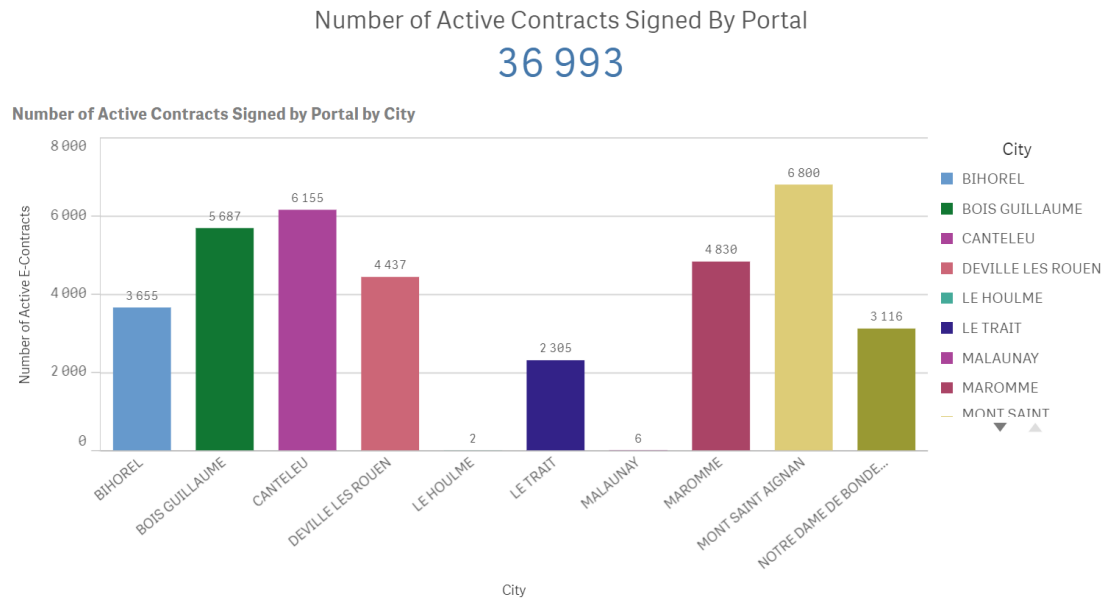


Figure 7 Example of representation for the of the *Number of Active Signed Contracts*

PPIs are usually modeled and calculated from a Non-Customizable Process. This implies losing the Process Variability perspective. If a process variant evolves, the PPI definitions in the Customizable Process Model will not be updated accordingly. On one hand the deployment of a performance management solution takes time and resources, which limits the PPI evolutions and increases the cost for organizations (Del-Río-Ortega et al., 2019). On the other hand, the significant gap that exists between PPI implementation languages and natural language may cause errors in PPIs evolutions. Additionally, PPI developers must manually detect and remove the ambiguities generated by natural language to properly calculate PPIs (Van der Aa et al., 2016). This is an error-prone task since PPI developers often do not share the same PPI definitions as decision-makers, due to the nature of their jobs. Indeed, developers are closer to technology, while decision-makers are closer to management (Del-Río-Ortega et al., 2019).

As we observed, Customizable Process Models and PPIs are usually modelled separately, especially when dealing with PPI Variability. There are different definitions and ways to calculate PPIs, which complicates PPI management. In most cases the modeling of PPI Variability does not have an explicit link with Customizable

Processes, which generates redundant models and makes PPI adjustment and PPI maintenance difficult.

This induces the following research questions:

- I) How to integrate PPIs into Customizable Process Models?
- II) How to model PPI Variability in the context of Customizable Process Models?
- III) How to recalculate an existing PPI or calculate a new PPI according to Customizable Processes?

1.3 GOALS AND CONTRIBUTIONS: CONVERGING PROCESS AND PPI VARIABILITY

Our objective is to bring innovative solutions for the modeling, the integration and the development of PPI Variability linked to Customizable Process Models. The object of our work is therefore to specify a language and a method to represent and manipulate models dedicated to PPI Variability linked to Customizable Process Models. The study of current PPI modeling solutions shows that most current approaches do not consider Customizable Process Models, nor consider a method to calculate or recalculate existing PPIs. Likewise, those approaches do not allow the calculation of PPIs.

The model for PPI Variability that we define extends those proposed in the literature with a conceptual model. This makes it possible to describe a model and calculate PPIs from a Customizable Process Model. This model incorporates a link with process activities to identify the necessary information to calculate useful PPIs for decision makers. The originality of our approach comes from the modeling of PPI Variability linked to Customizable Process Models, which is not treated in other works.

We propose an indicator calculation tree supported by a meta-model that integrates the Customizable Process Model concepts with PPI Variability. We specify a PPI Variability model relying on a PPI model, which is carried out through the design of indicators as a tree. Both the process model and the PPI model rely on the feature model approach to the quality of the link between activities and PPIs. We define a language to model PPI Variability, which has been validated through an experimental

evaluation. Our model constitutes a generalization of PPIs as interconnected features according to several design constraints. This model is the first contribution and constitutes the first step to develop our second contribution: a method for designing PPI Variability linked to Customizable Process Models. Finally, our third contribution is an executable software tool to build a PPI model and support the method that links them to the Customizable Process Model.

1.4 ORGANIZATION OF THE DOCUMENT

This thesis is organized in 8 chapters:

- I) In chapter 1, we introduce the thesis by giving the goals, the research context with a case study and the research problem. We presented the problem concerning the modeling of the Variability and the customization of business processes. We mentioned that according to our knowledge, PPI Variability has not been linked to Customizable Process Models. We raised the issue that an approach to help PPI Variability modeling and calculation is necessary to promote reusability in the context of a software publisher.
- II) In chapter 2, we carry out a state of the art concerning current research in the field of Customizable Process Models and PPI modeling to determine if those domains are connected and if PPI approaches support Variability. We show that the modeling and calculation of PPI Variability linked to Customizable Process Models is not studied in depth in the literature. We distinguish between the concepts of Key Performance Indicator and Process Performance Indicators. We also present the main solutions currently proposed in the domain of Customizable Processes to model process variants and current approaches to model indicators and Process Variability.
- III) In chapter 3, we present an overview of our approach, and its 3 main elements is presented: the Process Performance Indicator Calculation Tree (PPICT), the method for modeling PPI Variability integrated to Customizable Process Models and the software tool. The research methodology employed for our contribution is described.

- IV) In chapter 4, we present in more detail our first contribution: the Process Performance Indicator Calculation Tree (PPICT), which is a model designed to support the PPI Variability. We define the main concepts of our model: Process Performance indicator PPI, PPI reference, PPI variant, PPI Family, relations between PPIs, relations between PPIs and activities and the rules between PPI variant and PPI references. These rules are represented using relational algebra operators.
- V) In chapter 5, we present our second contribution: the Process Performance Indicator Calculation Method (PPIC Method), which is a conceptual method for modeling Process Performance Variability linked to Customizable Process Models. This method generalizes the analysis and performance evaluation of process variants. We detail each step using synthetic definitions and examples based on the presented case study: the *Create Contract process*.
- VI) In chapter 6, we specify the validation process used during the human-centered computer science research evaluation. We describe an experimental protocol of 10 qualitative interviews. We explain the PPI development improvements through these individual interviews of INCOM's engineers and novice students when using the PPICT Method.
- VII) In chapter 7, we describe the software tool carried out to implement our proposals. The purpose of this tool is to help to define and design PPI Variability linked to Customizable Process Models, based on the concepts that we have defined for the modeling and development of PPIs. It allows us to graphically and interactively produce a PPI model and to execute an indicator query.
- VIII) Finally, in chapter 8, we conclude the document, focusing on advantages, limitations and perspectives.

Chapter 2: Modeling PPI Variability and Customizable Process: State of Art

As mentioned in the first chapter, a business process is a set of interrelated tasks that end with the delivery of a service or product to a customer. Business process has also been defined as a set of activities and tasks that, when performed, will fulfill one of the business goals (OMG, 2016). The process should include clearly defined inputs and outputs. The inputs consist of all the factors that contribute (directly or indirectly) to the added value of a service or product. These processes can be categorized into management processes, operational processes, and supporting processes. To model and execute a business model, a set of methods and tools are used that analyze, optimize and control the implemented processes. These tools are called Business Process Management Systems (BPMS), and some examples are Bonita BPM², Camunda³ or Activiti⁴. As discussed in the first chapter, Process Performance Indicators (PPIs) have been developed to measure business process performance. PPIs represent the process goals that a company wants to achieve on an operational level. They can be measured directly by data generated within a specific process flow.

Definition: A *Customizable Process Model* is defined as a consolidated model of the possible variants of a process, where the variation points can be customized via transformations (La Rosa et al., 2017). A Customizable Process Model captures recurrent processes through a systematic modeling of variability and commonality (Ognjanovic et al., 2012). A Customizable Process Model could be illustrated using a business process modeling language for the representation of temporal and logical dependencies of activities in a business process.

Definition: *Process Performance Indicators (PPIs)* are defined as the performance perspective of a business process and are a particular case of *Key Performance Indicators (KPIs)*. KPIs as well as PPIs are a set of measures used to

² <https://fr.bonitasoft.com/>

³ <https://go.camunda.com/>

⁴ <https://www.activiti.org/>

evaluate organizational performance (Parmenter, 2015). PPIs facilitate decision-making and enable the identification of improvements of the process (Del-Río-Ortega et al., 2013). PPI management is part of the business process lifecycle. Indeed, PPIs are used to analyze the performance of the business processes (Estrada Torres et al., 2016).

The application of PPIs in Customizable Process Models complicates their modeling and management. When a process model is instantiated, the calculation and definition of a PPI can be different depending on the process variants. On the one hand, classical frameworks such as BPMN do not allow the modeling of Customizable Process, as we will see in section 2.1, where 5 approaches are analyzed to define Customizable Process Models. On the other hand, different approaches that model PPI Variability are proposed, but they focus on Non-Customizable Process Models (cf. section 2.2).

Hence, it is necessary to define new mechanisms to help PPI developers to identify and organize the essential information to model variable PPIs in Customizable Process Models. Current software publishers' needs are focused on the measurement of Customizable Process Model performances. The association between Business Process Variability and PPI Variability implies that PPIs should be redefined (Del-Río-Ortega et al., 2019), i.e., the PPIs should be redesigned and recalculated for the customized processes given a particular context. Section 2.3 discusses the challenges faced and shows which existing approach will be used for our proposition. INCOM's *Create Contract* process (presented in the previous chapter) will be used to illustrate the studied approaches from the literature as well as the interest of our proposition in this thesis.

2.1 CUSTOMIZABLE PROCESS MODELS

As mentioned before, Customizable Process Models capture the variants of a process family model by adding process fragments (cf. Figure 9) or deleting process fragments (cf. Figure 8). A Customizable Process Model manages customization decisions at design-time and/or run-time. For instance, during design-time, we seek to customize all process variants to be executed in a particular organizational setting. In comparison, during the run-time, we seek to customize one or a few process variants, i.e., only the process variants affected by the customization decisions.

2.1.1 Customizable Process Models requirements for PPI modeling

To model and easily implement PPIs in some Customizable Process Model approaches, we have identified 7 fundamental criteria divided in the following two perspectives:

- 1) **The integration level between PPIs and Customizable Process Models**, i.e., the capacity to include essential information to calculate a PPI as well as to provide the support to model a PPI for a known or an unknown process variant.
- 2) **The usability of the Customizable Process Model approaches**, i.e., the possibility to guide users to make appropriate use of the model and its tools.

Regarding **the perspective “integration level between PPIs and Customizable Process Models,”** the approach must:

- I) Support variability by restriction to model and calculate PPIs for known process variants.

Variability by restriction refers to a Customizable Process Model that contains all possible behaviors of all process variants (La Rosa et al., 2017). Customization is accomplished by restricting the behavior of the Customizable Process Model (La Rosa et al., 2017), e.g., activities may be skipped or blocked during customization (cf. Figure 8). Customizable Process Models managing variability by restriction are also called Configurable Process Models, which is defined as “the ability to express and produce different variants of a process from a configurable model” (Reichert et al., 2015). Some approaches that support this variability are PESOA (Schnieders & Puhlmann, 2006) and C-YAWL (Gottschalk et al., 2009), among others.

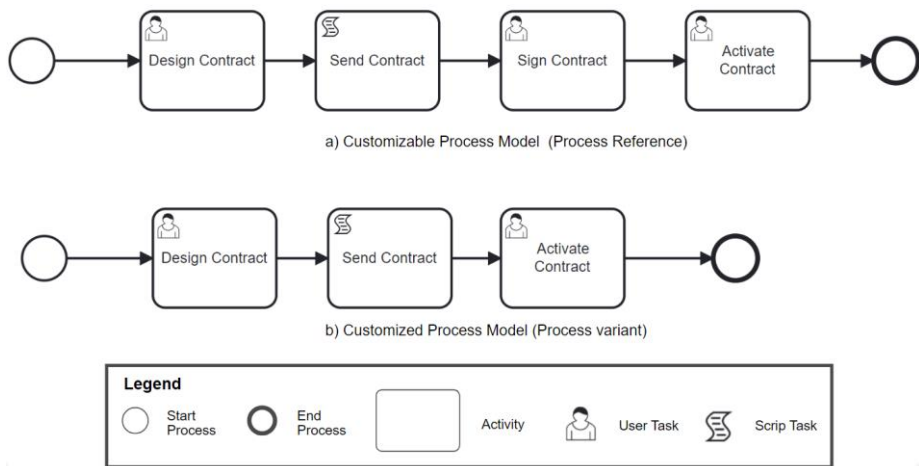


Figure 8 Example of variability by restriction compared to (a), in (b) the activity sign contract has been removed.

II) Support variability by extension to model and calculate PPIs for process variants not known a priori.

Variability by extension refers to a Customizable Process Model that does not contain all behaviors. Instead, it represents the most common behavior in process variants. During customization, the model's behavior is extended for a specific context (La Rosa et al., 2017), e.g., new activities may be inserted to create a dedicated variant: for example, in Figure 9 activities *Sign Contract by Mail* and *Sign Contract by Portal* have been added to the process variant. Some examples of approaches using this type of Process Variability are PROVOP (Reichert et al., 2015), and vBPMN (Döhning & Zimmermann, 2011) among others.

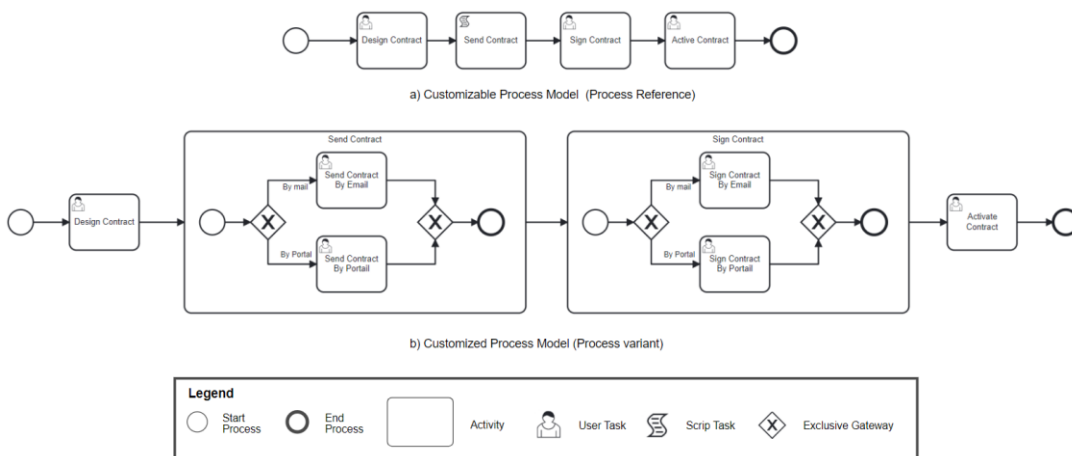


Figure 9 Example of variability by extension, in (b) activities Send Contract by Email, Send Contract by Portal, Sign Contract by Email and Sign Contract by Portal have been added compared to (a)

III) Support objects, specifically data-objects, to extract necessary information to calculate PPIs.

Data-objects represent the data integrated into the process, the data resulting from the process, the data to be collected and the data to be stored. The capacity to capture variability in data objects, e.g., data on the signed contract could be different depending on the contractual conditions.

IV) Have an executable and extensible implementation to include the calculation of variable PPIs.

Executable and extensible implementation refers to resolving inconsistencies between activities and data object associations.

For the perspective “**usability of the Customizable Process Model approaches**”, the approach must:

V) Be validated through a real case study, preferably in a software publisher case which is our use case.

Validation refers to implementation using a real-life Customizable Process Model through discussions with experts.

VI) Help avoid behavioral anomalies in particular deadlocks and livelocks when the Customizable Process Model is instantiated.

Help avoid behavioral anomalies refers to always having the possibility to complete any customized process model properly without deadlocks or livelocks. The model could be for instance based on Petri-nets relying on the definition of soundness (Aalst et al., 2010). Customizable Process Models using Petri-nets execute algorithms to ensure that a customized process model is free of behavioral anomalies such as lack of synchronization and deadlocks.

VII) Guide users as much as possible when making customization decisions.

Guide users refers to the selection of one customization process model option or another to prevent users from making inconsistent decisions of process model customizations from the domain angle.

Relying on the aforementioned criteria, we propose an analysis of the more relevant approaches that model customizable processes. These approaches have been chosen according to partial or total compliance with at least 5 of the 7 proposed criteria.

2.1.2 Configurable integrated Event-driven Process Chains (C-iEPCs)

One of the first approaches proposed by (La Rosa et al., 2011) was the C-iEPCs (Configurable integrated Event-driven Process Chains), which is an extension of the Event-driven Process Chains (EPC) language, a business modeling language to represent the logical and temporal dependencies between business process's activities. An iEPC is an Event-driven Process Chains with resources objects assigned to activities. A C-iEPC model captures the common elements of an iEPC family (La Rosa et al., 2011). Configurable nodes are used to model the differences between the process variants. A configurable node contains a set of process customization options, i.e., each set refers to one or more process variants. To model such customization, the behavior of the C-iEPC must be restricted to each configurable node, using a customization option. In this approach, once the C-iEPC is modeled, the model is transformed into an iEPC to eliminate the options that are not necessary. Then, the original variants of the process family can be modified.

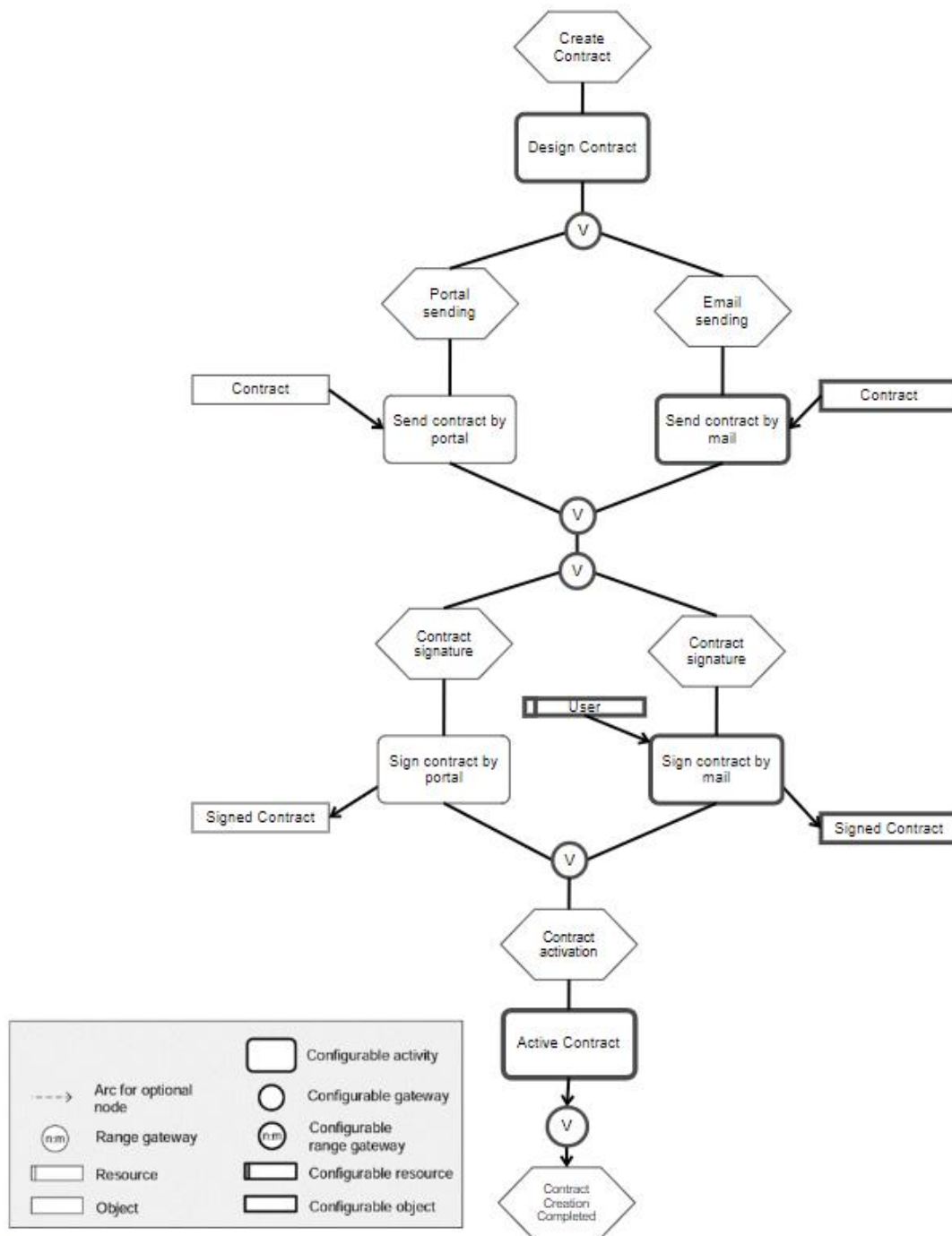


Figure 10 Example of the C-iEPC model based on the *Create Contract* Process

Figure 10 shows an example of a C-iEPC model. Model links can be customized for restrictive or non-restrictive variants, e.g., a configurable OR can be left as a normal OR, i.e., without restriction, or restricted to a XOR or even an AND. A configurable activity can be kept ON or OFF, i.e., the activity can be hidden in the custom model. This enables the possible choice of keeping the activity or not until the moment of the

execution. In more recent versions of this approach, it is even possible to model resources (known as roles in the C-iEPC model) as well as objects.

Configurable input objects can be restricted for use, so they are not destroyed by the activity after use. It is possible to derive an iEPC from a C-iEPC, if the C-iEPC is structurally correct. The correctness of the behavior is guaranteed by the inference of constraints. The stakeholder’s criteria are revealed during the personalization through a questionnaire, which captures the properties and their values using the configurable nodes (La Rosa et al., 2017). This approach allows the customization of CiEPC through questionnaire templates to obtain the configured process. These templates are compatible with the Synergia toolkits (La Rosa & Gottschalk, 2009), (Dechow et al., 2005). The use of CiEPCs has been validated through a case study of pre-production of films (La Rosa et al., 2011). But the C-iEPC does not provide any execution support.

Table 1 presents our requirements to model Customizable Process Models linked to variables PPIs and summarizes the evaluated requirements of the C-iEPC approach:

Table 1 Requirements analysis of the C-iEPC approach

Requirement	C-iEPC approach
I) Support variability by restriction to model and calculate PPIs for known process variants.	<i>C-iEPC</i> supports this requirement because it has configurable nodes used to model the differences between the process variants. A configurable node could contain both a set of process customization options and PPI customization options.
II) Support variability by extension to model and calculate PPIs for process variants, which are not known a priori.	<i>C-iEPC</i> does not support this requirement because it does not allow adding configurable nodes for process variants which are not known a priori. PPI Variability by extension is impossible
III) Support objects, specifically data-objects, to extract necessary information to calculate PPIs.	<i>C-iEPC</i> supports this requirement because it supports configurable input objects, which can be restricted for use, so they are not destroyed by the activity after use.

IV) Have an executable and extensible implementation to include the calculation of variable PPIs	C-iEPC does not support this requirement because it does not provide any execution support.
V) Be validated through a real case study, preferably in a software publisher case.	C-iEPC supports this requirement because it has been validated through a case study of pre-production of films.
VI) Avoid behavioral anomalies in particular livelocks and deadlocks when the Customizable Process model is instantiated.	C-iEPC supports this requirement because the correctness of the behavior is guaranteed by the inference of constraints.
VII) Guide users as much as possible when making customization decisions to select one option or another.	C-iEPC meets this requirement because it guides users through questionnaire templates to obtain the customized process model.

2.1.3 Configurable Workflows

The approach *Configurable workflows* (Gottschalk et al., 2008) was proposed as a conceptual model and then as executable language. One of the most representative extensions of YAWL (Yet Another Workflow Language) is the Configurable YAWL (C-YAWL) (Gottschalk et al., 2008). YAWL handles split and join links linked to activities: a join heads an activity to model the activity's behavior. C-YAWL proposes ports to identify configurable links which are represented with a wider border. Configurable splits have one *outflow* port for each flow combination. In the same way, configurable joins have one *inflow* port for each flow combination. Figure 11 (a) shows a C-YAWL example for a port customization, where XOR-Split is used to route the process flow as well as XOR-join which can be activated by the incoming flows *Ia* and *Ib*.

In YAWL, an activity with a single outgoing or incoming flow has an implicit XOR behavior (Gottschalk et al., 2008). This behavior is shown graphically if the gateway must be made configurable. Likewise, it is possible to hide this activity inflow port.

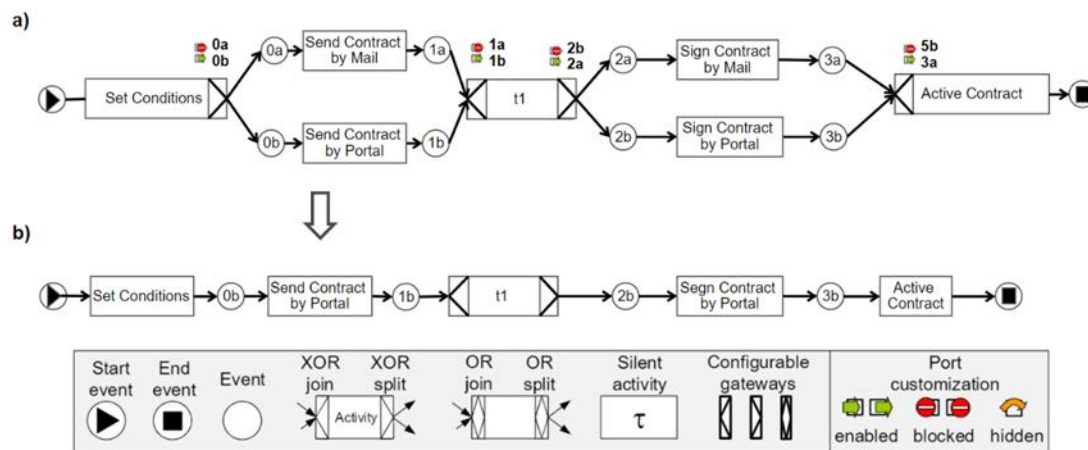


Figure 11 (a) Example of create contract process in C-YAWL with a sample customization; (b) the customized create contract process model.

YAWL has hide and lock operations that can also be applied to the next elements such as compound activities, and multi-instance activities. Figure 11(b) shows the YAWL model resulting from the customization of the example, after applying the transformation algorithm defined in (Gottschalk et al., 2008). This algorithm removes all nodes after customization to guarantee the structural correctness of the model. Likewise, conflicts in data conditions are resolved, to guarantee the executability of the model. To ensure correct behavior for custom models, two techniques are proposed: one relying on constraint inference and the other relying on partner synthesis. These techniques rely on the application of hiding and blocking using a Petri net to model BP (Aalst et al., 2010). This net is based on the definition of soundness to ensure that a customized process model is free of behavioral anomalies such as a lack of synchronization and deadlocks. Questionnaire templates are used to provide decision support using Synergia toolset (La Rosa & Gottschalk, 2009). YAWL has been formalized (Gottschalk et al., 2008) and implemented in YAWL Editor to create, customize, and transform C-YAWL models into YAWL models. The use of C-YAWL models with questionnaire models has been validated in the municipal domain (Lönn et al., 2012) (La Rosa & Gottschalk, 2009).

Table 2 presents our requirements to model Customizable Process Models linked to variable PPIs and summarizes the evaluated requirements of the *Configurable workflows* approach:

Table 2 Requirements analysis of the Configurable Workflows approach

Requirement	Customizable Workflows approach
I) Support variability by restriction to model and calculate PPIs for known process variants.	The <i>Customizable Workflows</i> approach supports this requirement because it has customizable ports to enable, block or hide an execution way. A configurable port could contain both a set of process customization options and PPIs customization options.
II) Support variability by extension to model and calculate PPIs for process variants, which are not known a priori.	The <i>Customizable Workflows</i> approach does not support this requirement because it does not allow adding configurable ports for process variants which are not known a priori.
III) Support objects, specifically data-objects, to extract necessary information to calculate PPIs.	The <i>Customizable Workflows</i> approach does not support this requirement because it does not support data-objects.
IV) Have an executable and extensible implementation to include the calculation of variable PPIs	The <i>Customizable Workflows</i> approach supports this requirement because it has been formalized (Gottschalk et al., 2008) and implemented in the YAWL Editor.
V) Be validated through a real case study, preferably in a software publisher case.	The <i>Customizable Workflows</i> approach supports this requirement because it uses the C-YAWL model which has questionnaire models that have been validated in the municipality domain (Gottschalk et al., 2009).
VI) Avoid behavioral anomalies in particular livelocks and deadlocks when the Customizable Process model is instantiated.	The <i>Customizable Workflows</i> approach supports this requirement because to ensure the correct behavior for custom models, two techniques are proposed: one relying on constraint inference and the other one relying on partner synthesis.
VII) Guide users as much as possible when making customization decisions to select one option or another.	The <i>Customizable Workflows</i> approach supports this requirement because it uses the C-YAWL model which has questionnaire models guiding users to make customization decisions.

2.1.4 *Template and rules*

The approach *Template and rules* (Kumar & Yao, 2009) represents the variability of a process family by including a set of rules linked with a business process template. The business process template refers to the Customizable Process Model: a simple, block-structured process model that must be chosen to have the shortest structure of all the process variants in a family. Rules are sequences of operations that are used to customize the template by restricting or extending its behavior. Change operations affect the flow of control (deleting, inserting, moving, or replacing a fragment), resources (assigning a resource to an activity or changing the value of a resource property), and data objects (assigning a value to a data attribute, or changing the input / output data value of an activity). The operations on resources and objects are modified, while those on the control flow are a combination of delete and insert (La Rosa et al., 2017). Gateways cannot be directly customized and set points are not explicitly rendered, meaning that change operations can be applied to any chunk of process model. As a result, however, in *Template and Rules*, the variability is not represented in any process model perspective and can only be inferred from the rules that accompany the template.

The Template and rules conditions provide a Customizable Process Model abstraction. Figure 12 shows a running example of the *Template and rules* notation using the BPMN language.

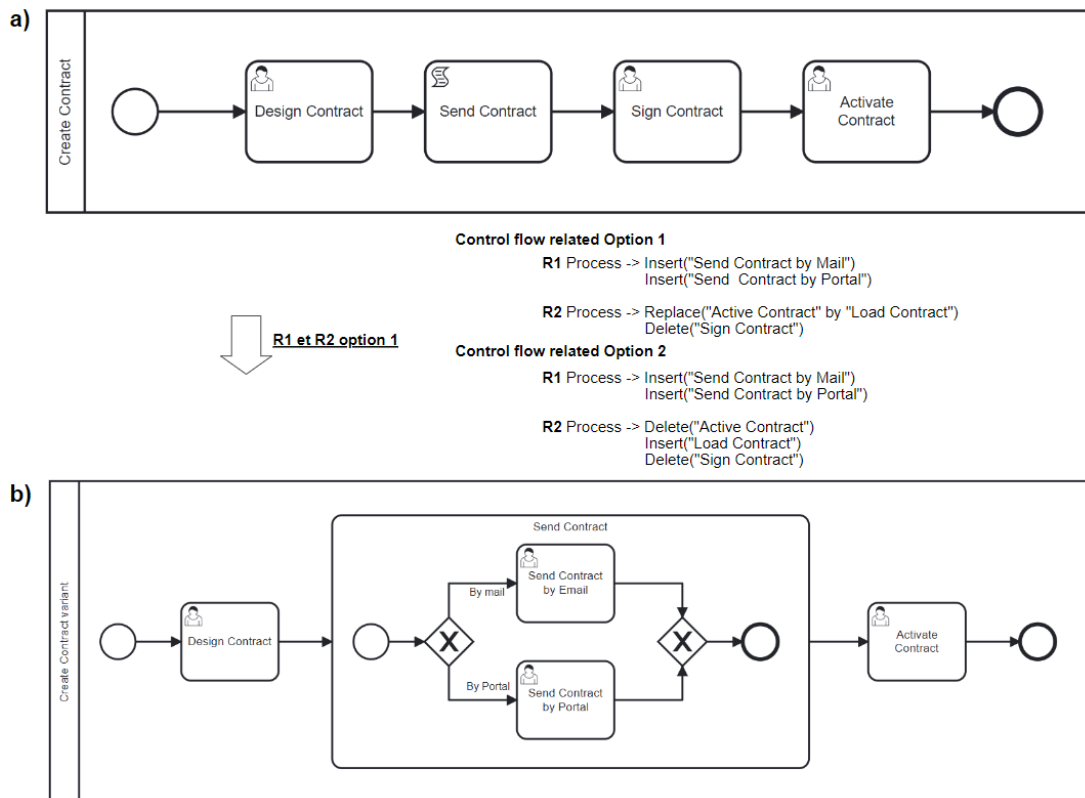


Figure 12 (a) The Create Contract Process Reference in *Template & Rules*. (b) Customized model of the Create Contract Process

The *Template and rules* approach is implemented using The BPEL language and uses a set of rules based on the Drools-expert rule engine (Shigarov, 2015). Thus, this approach can check for conflicts between available rules, e.g., delete an activity when another rule is seeking to add a new one. Each rule is checked to make sure that it is not possible to delete an inexistent node. Finally, a customized process model, i.e., a process variant, is obtained relying on those rules that are applicable and non-conflicting. The customized process model is checked for data inconsistencies, e.g., when the data input is no longer available. This check is done to guarantee the customized model executability. Nevertheless, the approach has not been validated in practice.

Table 3 presents our requirements to model Customizable Process Models linked to variables PPIs and summarizes the evaluated requirements of the *Template and rules* approach:

Table 3 Requirements analysis of the *Template and rules* approach

Requirement	<i>Template and rules</i> approach
I) Support variability by restriction to model and calculate PPIs for known process variants.	The <i>Template and Rules</i> approach supports this requirement because the rules are a sequence of replace, delete and assign or change a resource to customize the template by restricting its behavior.
II) Support variability by extension to model and calculate PPIs for process variants, which are not known a priori.	The <i>Template and Rules</i> approach supports this requirement because the rules are a sequence of insert, replace, delete and assign or change a resource operation to customize the template by extending its behavior.
III) Support objects, specifically data-objects, to extract necessary information to calculate PPIs.	The <i>Template and Rules</i> approach partially supports this requirement because it offers the possibility to assign or change the value of an input or output data.
IV) Have an executable and extensible implementation to include the calculation of variable PPIs	The <i>Template and Rules</i> approach partially supports this requirement because it has been implemented using the language BPEL. Nevertheless, this tool is not publicly available.
V) Be validated through a real case study, preferably in a software publisher case.	The <i>Template and Rules</i> approach does not support this requirement because it has not been validated in practice.
VI) Avoid behavioral anomalies in particular livelocks and deadlocks when the Customizable Process model is instantiated.	The <i>Template and Rules</i> approach supports this requirement because to guarantee the executability of the customized model, it uses only those rules that are non-conflicting to check the data-flow inconsistencies.
VII) Guide users as much as possible when making customization decisions to select one option or another.	The <i>Template and Rules</i> approach does not meet this requirement because it has no guidance support.

2.1.5 Business Process Feature Model

The Business Process Feature Model (BPFM) (Cognini et al., 2016) combines concepts coming from both feature modeling and from BP modeling. A Feature Model

is a graphical model that uses a tree representation. This notation clearly shows the relationships among features. The root of the tree represents the generic product (the family). BPFM is an extension of Feature Models incorporating business process aspects. It represents the activity building blocks, such as atomic tasks and complex sub-processes. BP models can be composed of three perspectives; the *behavioral perspective* that represents the dynamic behavior of an executable BP model; the *information perspective* representing data objects involved in the BP; and the *operational perspective*, which represents the implementation details of the BP activities. BPFM includes constraints between the static inclusion and the dynamic occurrence of an activity. With reference to the information perspective, the notation supports the part of relation in data object modeling. While some data objects are primitive, others can be decomposed into more fine-grained objects.

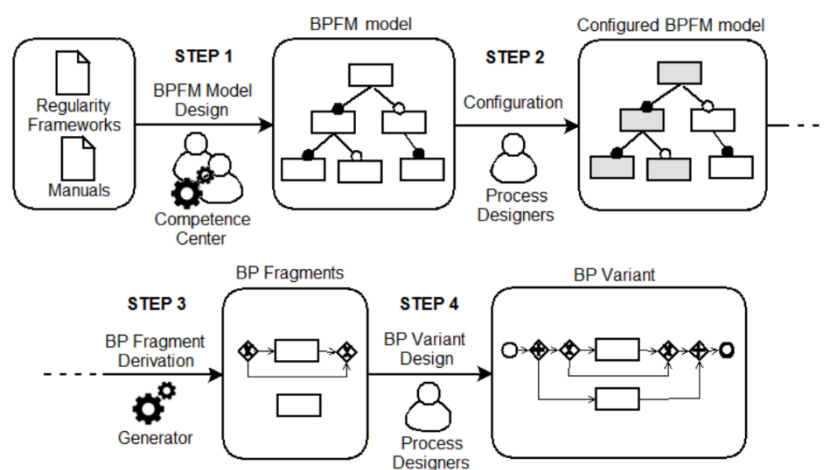


Figure 13 BPFM Method (Cognini et al., 2016)

The BPFM allows the modeling of all variants of a reference process in the case of a business process family (step 1 of Figure 13) and via a configuration step (step 2), it is possible to derive the most suitable one for the specific organization. In step 3, BPFM generates the BPMN fragments of the business process variant, i.e., the BPMN fragments of the customized process model. For this generation, BPFM executes several algorithms to ensure that the fragments of the customized process model are free of behavioral anomalies such as deadlocks. In step 4, BP designers could derive the BP fragments to enrich them with control flow information relying on the BPMN proposals to avoid livelocks. At this stage, we must consider the characteristics of an organization and adapt our processes according to the precise rules of each

organization that we could not know before the configuration of the processes (Variability by extension). Knowing and modeling all the possible variables of a process is not easy, even more so especially when extensions of the reference processes are managed.

BPFM has been implemented on ADOxx⁵ development platform (cf. Figure 14).

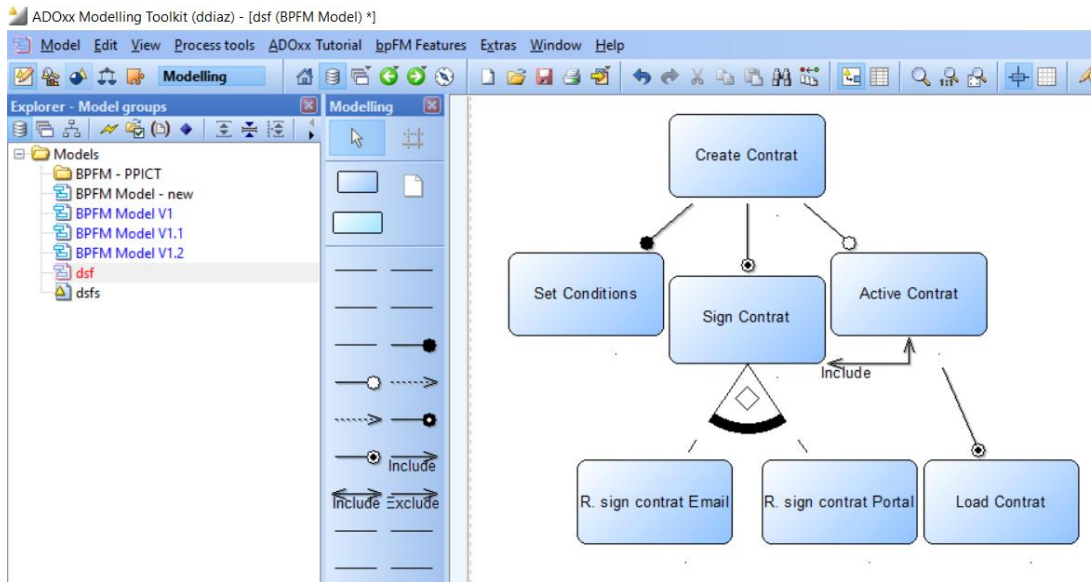


Figure 14 A BPFM Model created via the BPFM tool.

A BPFM model is composed of activities organized in a tree (cf. Figure 14) where the root activity identifies the family of business processes. BPFM has diverse levels of specifying a process where each internal activity that is not a leaf in the tree describes a sub-process. On the other hand, the leaf activities of the tree represent atomic activities, i.e., tasks. BPFM details all tasks using the same symbols proposed by BPMN 2.0. The relationships between activities are called constraints which allow a representation of the variability in two dimensions:

1. If each supplementary activity must be inserted in each variant of the process. This means that the activity must be selected during configuration.
2. if activities must or can be executed at runtime. The goal is to customize both static and dynamic activities after deploying the process. The type

⁵ <https://www.adoxx.org/>

of restrictions can be binary or multiple depending on the relationships between secondary activities and main activities.

Table 4 presents our requirements to model Customizable Process Models linked to variables PPIs and summarizes the evaluated requirements of the *BPFM* approach:

Table 4 Requirements analysis of the BPFM approach

Requirement	BPFM approach
I) Support variability by restriction to model and calculate PPIs for known process variants.	The <i>BPFM</i> approach supports this requirement because it allows modifying and deleting process elements during the configuration phase.
II) Support variability by extension to model and calculate PPIs for process variants, which are not known a priori.	The <i>BPFM</i> approach supports this requirement because it allows adding business process elements after the configuration phase.
III) Support objects, specifically data-objects, to extract necessary information to calculate PPIs.	The <i>BPFM</i> approach supports this requirement because it supports configurable input and output objects, which can be restricted or added during the customization phase.
IV) Have an executable and extensible implementation to include the calculation of variable PPIs	The <i>BPFM</i> approach supports this requirement because it proposes a conceptual model, but also an exploitable tool based on ADOxx platform.
V) Be validated through a real case study, preferably in a software publisher case.	The <i>BPFM</i> approach supports this requirement because it has been validated through a case study in public administrations.
VI) Avoid behavioral anomalies in particular livelocks and deadlocks when the Customizable Process model is instantiated.	The <i>BPFM</i> approach supports this requirement because the correctness of the behavior is guaranteed when generating the process variant. BPFM executes several algorithms to ensure that the customized process model be free of behavioral anomalies such as deadlocks. These algorithms are used also to generate the customized process model on BPMN, helping to avoid livelocks.

VII) Guide users as much as possible when making customization decisions to select one option or another.	The <i>BPFM</i> approach partially supports this requirement because it guides users through automatic pop-up execution for the constraints that are not validated but it does not have verification questionnaires.
---	--

2.1.6 Synthesis of Customizable Process Models

After evaluating the approaches regarding our desired conditions shown previously, we note that none of them meets all the requirements and constraints except the Business Process Feature Model (BPFM) (Cognini et al., 2016). None of them integrates the PPI Variability modeling (La Rosa et al., 2017), (Estrada-Torres et al., 2016) (cf. Table 5). BPFM includes refinement variants even if the process has been customized. Therefore, we can consider PPI modeling for Process Variability by restriction and by extension. BPFM also considers the deployment context information adapting execution paths for every process variant. BPFM has been implemented in the ADOxx platform, which allows it to be executable, to guide users to make customization decisions and to be easily modified to include PPI modeling. Moreover, when generating the process variant, BPFM executes several algorithms to ensure that the customized process model is free of behavioral anomalies such as deadlocks. These algorithms are also used to generate the customized process model, i.e., process variant, on BPMN. Thus, helping to avoid livelocks. Likewise, BPFM supports data-objects making it possible to obtain information generated by each activity, which serves as a guide to calculate the desired PPIs. Finally, this approach has been validated considering several Public Administration scenarios through the European Project Budget Report case study endorsed by the Learn Pad Project⁶.

After having analyzed the approaches that allow the modeling of Customizable Process Models, we will analyze the approaches that allow us to model the Process Performance Indicators. As with the analysis of the Customizable Process Models, we will propose classification criteria that will be used to evaluate each approach.

⁶ <http://www.learnpad.eu/>

Table 5 Evaluation of Customizable Process Model Approaches

	Reference	The integration level between PPIs and Customizable Process Models				The usability of the Customizable Process Model approaches		
		I) Support variability by restriction to model and calculate PPIs for known process variants.	II) Support variability by extension to model and calculate PPIs for process variants, which are not known a priori.	III) Support objects, specifically data-objects, to extract necessary information to calculate PPIs.	IV) Have an executable and extensible implementation to include the calculation of variable PPIs.	V) Be validated through a real case study, preferably in a software publisher case.	VI) Avoid behavioral anomalies in particular livelocks and deadlocks when the Customizable Process model is instantiated.	VII) Guide users as much as possible when making customization decisions to select one option or another.
Configurable integrated Event-driven Process Chain (C-iEPCs)	(La Rosa et al., 2011)	Yes	No	Yes	No	Yes	Yes	Yes
Customizable Workflows (Language: C-YAWL, C-SAP, C-BPEL)	(Gottschalk et al., 2008)	Yes	Yes	Partially	Partially	Yes	No	No
Template and rules (Language: Block-Structured BPEL)	(Kumar & Yao, 2009)	Yes	Yes	Partially	Partially	Yes	No	Yes
Business Process Feature Model (BPFM)	(Cognini et al., 2016)	Yes	Yes	Yes	Yes	Yes	Yes	Partially

2.2 PROCESS PERFORMANCE INDICATOR MODELING APPROACHES

In this subchapter we provide definitions of basic concepts for the analysis and evaluation of different approaches according to a list of requirements defined from our experience. Each requirement will be detailed and explained. Then each approach will be analyzed and finally a summary of the evaluation carried out will be given.

Some models such as ARIS (Davis & Brabander, 2007) use Key Performance Indicators to model cause-and-effect relationships aligned with organizational strategic goals. ARIS is an approach that enables companies to measure and to analyze the performance and structure of their business processes. An example of ARIS in the context of medical distribution is shown in Table 6. The ARIS approach makes it possible to conceptually identify the link between business process and performance indicator.

Definition: Key Performance Indicators (KPIs) are defined as a set of measures used to evaluate organizational performance (Parmenter, 2015). They are essential for managers to understand whether their businesses are being successful or need improving. (Marr, 2012). A KPI highlights areas that need attention as well as the performance of those areas that are going in the right way. KPIs are usually modeled separately from the business process. KPIs are also generic metrics applicable to any process, such as process duration, number of executions, or even process-specific measures. KPIs are based on the properties of business process's objects and they can be calculated for one single business process instance or they may be aggregated over several business process instances using for examples average, minimum or maximum aggregations.

To evaluate the performance requirements of business processes, Process Performance Indicators (PPIs) (Del-Río-Ortega et al., 2013) are used as a particular case of KPIs. PPIs permit managers to evaluate the effectiveness and efficiency of a business process (Del-Río-Ortega et al., 2016). In addition, PPIs can be measured directly by the data generated during the process flow (Rosenberg et al., 2011).

Table 6 Example of ARIS tool in the context of medical distribution

N°	Risks	KPI	Interest	Organizational unit	Actor
1	Unchecked demand of medicines	daily demand	know the real demand	care unit	head nurse
2	Varying demand of medicines	items in low level	control the real	pharmacy	pharmacy dispenser
		items in high level	consumption of medicines		
3	Inaccurate inventory	inventory gap	insure the inventory reliability	pharmacy	pharmacy dispenser
4	Lack of verification during the reception of parts	reception quality rate	insure the reception reliability	pharmacy	pharmacy dispenser
5	lack of delivery monitoring	service rate of supply division	control the delivery delay	pharmacy	head pharmacist
6	Irregular replenishment of unit care	medecines advance rate	control the advance reason	pharmacy	pharmacy dispenser

Process Performance Indicators (PPIs) are generally defined in an informal way, e.g., in natural language, which leads to problems of ambiguity, coherence and traceability in relation to Process Models (Del-Río-Ortega et al., 2013), e.g., missing information in the definition of a PPI. PPIs are usually defined from a Non-Customizable Process or instance losing the perspective of the Customizable Process Model.

The PPI Variability has not been addressed so extensively in the literature (Estrada-Torres et al., 2016) (Del-Río-Ortega et al., 2013). For instance, classical architectures such as Data Warehouse, Business Intelligence, Business Activity Monitoring (Friedenstab et al., 2012), Business Performance Management (Golfarelli et al., 2004) or Performance Indicator Modeling (Popova & Sharpanskykh, 2010) deal with the importance of enforcing goals defined by business strategies. Likewise, they deal with metrics to model and calculate indicators. Nevertheless, in the case of Customizable Process Models, information extraction from business data is not enough, especially when different PPI definitions depend on flexible evaluation criteria and process variants.

In the next section, we will talk about our 7 requirements to analyze and evaluate the PPI approaches.

2.2.1 Process Performance Indicator Requirements

To model variable PPIs linked to Customizable Process Models, we have identified 7 requirements necessary to analyze current performance indicator models. These requirements were determined based on our experience and the experience of other authors as (Estrada-Torres et al., 2021), They are described individually for each approach presented in the Table 12. They are divided into two groups, in an equivalent structure to that of the previous analysis:

Group 1: **The expressive capacity of PPIs.** For this group, the approach should make it possible to:

- I) Define a variable performance indicator.
This requirement seeks to assess the ability to define PPI Variability and seeks to establish how it impacts business process evaluations, and how to deal with variable PPIs.
- II) Model the relation between PPIs.
This requirement seeks to evaluate the ability to model the relationship between PPIs, to know how PPIs are related to each other and if there is more than one type of relationship, e.g., one-to-one or one-to-many, and if there is more than one category of relationships, e.g., direct, or indirect relationship.
- III) Consider PPI Variability by restriction.
This requirement seeks to evaluate the ability to model the PPI Variability when whose PPI variants are known a priori, that is, when the PPI template has all the known customization possibilities.
- IV) Consider PPI Variability by extension.
This requirement seeks to evaluate the ability to model the PPI Variability when whose PPI variants are not known a priori, that is, when the PPI template does not have all the known customization possibilities.

Group 2: The **integration level between PPIs and Customizable Process Models**. For this group, the approach should make it possible to:

- V) Model PPIs independently of the language used to model the BP (if this requirement is fulfilled, requirement VI must not be fulfilled).

This refers to the capacity to model PPIs based on a language different than the language to model Business Process. The goal is to be able to separate the models and draw a link between the business processes and the decision-making systems afterwards.

- VI) Model PPI Variability together with Customizable Process Models (if this requirement is fulfilled, requirement V must not be fulfilled).

This refers to the capacity to design the PPI Variability and Customizable Process in a single model to create a synergy between Customizable Processes Models and Process Performance Indicators.

- VII) Model the relation between PPIs and Activities of a business process.

This refers to the capacity to model the link between PPIs and the activities to identify the resources that generate the data used to calculate indicators.

2.2.2 Business Performance Measurement approach

Business performance measurement approaches provide tranches of management interest reports. Business performance frameworks and measurement methodologies offer little information regarding certain aspects such as quality or involvement level. Performance measurement models are instrumental in achieving excellence in BP when it comes to integrating and balancing stakeholders' needs. The most renowned business performance model is the Balanced Scorecard (BSC) (Kaplan et al., 1992). The Balanced Scorecard considers financial and non-financial indices. The BSC enables managers to discern organizational performance gaps linked to their

business strategy. BSC measures to improve business performance measurement and promote indicator quality. Nevertheless, the performance measurement models do not consider Customizable Process Models nor Performance Indicators Variability (Rahim et al., 2018) (Landström et al., 2018).

Table 7 presents our requirements to model variable PPIs linked to Customizable Process Models and summarizes the evaluated requirements of the Business Performance Measurement BSC approach.

Table 7 Requirements analysis of the Business Performance Measurement BSC approach

Requirement	Business Performance Measurement BSC approach
I) Define a variable performance indicator	The <i>Business Performance Measurement</i> approach does not support this requirement because it concerns only non-variable performance indicators.
II) Model the relationships between PPIs	The <i>Business Performance Measurement</i> approach supports this requirement because it allows the modeling of relationships between PPIs but not in the context of PPI Variability.
III) Consider PPI Variability by restriction,	The <i>Business Performance Measurement</i> approach does not support this requirement because it does not consider the diverse ways to calculate PPIs.
IV) Consider PPI Variability by extension	The <i>Business Performance Measurement</i> approach does not support this requirement because it does not consider PPI Variability.
V) Model PPIs independently of the language used to model the BP	The <i>Business Performance Measurement</i> approach supports this requirement because it allows the modeling of PPI in natural language and sparsely using BP modeling language.
VI) Model PPI Variability together with Customizable Process Models	The <i>Business Performance Measurement</i> approach does not support this requirement because it is designed to model indicators and not customizable processes.

VII) Model the relation PPI-Activities of a business process.	The <i>Business Performance Measurement</i> approach does not support this requirement because there is not a concrete link between PPIs and business process elements.
---	---

2.2.3 Business Activity Monitoring and Business Process Execution Measurement Model

Business Activity Monitoring (BAM) is defined as a tool for analyzing real-time business performance indicators to make better decisions about the effectiveness of operations. BAM seeks to minimize the decision-making latency by providing real-time information to take an adequate action (Friedenstab et al., 2012). BAM uses aggregation and quantitative measures in real-time of basic events to monitor and control business processes and know the current state of business processes. BAM relies on Key Performance Indicators (KPIs), which are typically used to measure business goals. Performance information provided by BAM systems is targeted at decision-makers. BAM applications are usually dashboards which display KPIs with the most essential information needed to achieve or monitor one or more goals. BAM is divided into 3 stages:

- I) The modeling of the business processes.
- II) The specification of event processing for monitoring and controlling the current state of business processes.
- III) The design of dashboards and KPIs.

Regarding expressiveness of PPIs modeling, (Delgado et al., 2014) propose the Business Process Execution Measurement Model (BPEMM), which is based on an existing software measurement ontology. The BPEMM provides a set of execution measures for business processes according to a continuous improvement in an organization. Having a predefined set of execution measures will help organizations to focus on the evaluation of selected aspects of BP execution, preventing them from spending valuable time in defining the execution measures by themselves. Moreover, BPEMM does not consider the definition of domain-specific and user-defined PPIs.

Table 8 our requirements to evaluate PPI modeling approaches and summarizes the evaluated requirements of the Business Process Execution Measurement Model.

Table 8 Requirements analysis of the BPEMM approach

Requirement	Business Process Execution Measurement Model
I) Define a variable performance indicator	The <i>Business Process Execution Measurement Model</i> approach does not support this requirement because it relies on performance indicators but not on variable performance indicators.
II) Model the relation between PPIs,	The <i>Business Process Execution Measurement Model</i> approach supports this requirement because it allows users to model relations between PPIs at a strategic level.
III) Consider PPI Variability by restriction	The <i>Business Process Execution Measurement Model</i> approach does not support this requirement because the indicators are modeled for a particular process. Therefore, variability by restriction is not considered in this approach.
IV) Consider PPI Variability by extension	The <i>Business Process Execution Measurement Model</i> approach does not support this requirement because the indicators are modeled for a particular process. Therefore, variability by extension is not considered in this approach.
V) Model PPIs independently of the language used to model the BP	The <i>Business Process Execution Measurement Model</i> approach supports this requirement because on one hand, it allows to model PPIs relying on a language based on measure aggregations and target definition. Likewise, it allows to model processes using BPMN.
VI) Model PPI Variability together with Customizable Process Models	The <i>Business Process Execution Measurement Model</i> approach does not support this requirement because the link with processes is based on an instance process and not on a process template.
VII) Model the relation PPI-Activities of a business process.	The <i>Business Process Execution Measurement Model</i> approach supports this requirement

	because it considers the link between process activities and indicators.
--	--

2.2.4 Data Warehouse Architecture

The data warehouse modeling concerns especially data stores. Indeed, the multidimensional approach represents the data in accordance with decision-makers' criteria, according to the different lines of possible analysis. For example, the data warehouse architecture proposed by (Ngo et al., 2019) combines non-SQL data bases applying the following capabilities: flexible schema, data integration from real multi datasets, data science and business intelligent support, high performance, high storage, security, governance and monitoring, replication and recovery, consistency, availability, distributed and cloud deployment

The multidimensional model consists of facts containing the measures to be analyzed and dimensions containing the parameters of the analysis. Within each dimension, parameters are organized hierarchically into levels of detail. This multidimensional representation of data induces new operations related to aggregations. These basic manipulation operations are linked to the structure and to the level of observation of values, called level of granularity of data.

Table 9 focuses on our requirements to model variable PPIs linked to Customizable Process Models and evaluates the requirements of the Data Warehousing approach.

Table 9 Requirements analysis of the Data Warehouse Architecture approach (Ngo et al., 2019)

Requirement	Data Warehouse Architecture
I) Define a variable performance indicator	The <i>Data Warehouse Architecture</i> does not support this requirement because it relies on data modeling for Non-Customizable decision-making systems.
II) Model the relation between PPIs,	The <i>Data Warehouse Architecture</i> does not support this requirement because it relies on a set of indicators but not on individual PPIs.

III) Consider PPI Variability by restriction,	The <i>Data Warehouse Architecture</i> does not support this requirement because it does not consider variability modeling. Therefore, we cannot customize a data warehouse depending on the business context.
IV) Consider PPI Variability by extension	The <i>Data Warehouse Architecture</i> does not support this requirement because it does not consider variability modeling. Therefore, we cannot customize a data warehouse for not supporting indicator variants.
V) Model PPIs independently of the language used to model the BP	The <i>Data Warehouse Architecture</i> partially supports this requirement because it is possible to model a set of indicators, but they are not linked to a Customizable Process Model.
VI) Model PPI Variability together with Customizable Process Models	The <i>Data Warehouse Architecture</i> does not support this requirement because Customizable Process modeling is separated from data warehouse modeling.
VII) Model the relation PPI-Activities of a business process.	The <i>Data Warehouse Architecture</i> does not support this requirement because PPIs Are neither linked with process nor with activities.

2.2.5 Business Intelligence Approach

Business intelligence systems blend data with analytical tools to present to decision makers only information relevant to their business activities. (Negash & Gray, 2008). Business Intelligence allows the understanding of the capabilities available in an organization. Business Intelligence uses both structured and semi-structured data. It seeks to help managers to make decisions regarding operations looking backwards to analyze descriptively and diagnostically the organization's performance (Velu, 2021).

The Business Intelligence Approach (BIA) (Li et al., 2008) proposes appropriate technologies along with each phase of the process to help management in implementation, which includes a decision support system that enables them to integrate various methodologies together.

Table 10 focuses on our requirements to model variable PPIs linked to Customizable Process Models and evaluates the requirements of the Business Intelligence Approach.

Table 10 Requirements analysis of the Business intelligence BIA approach

Requirement	Business intelligence Approach
I) Define a variable performance indicator.	The <i>Business Intelligence Approach</i> does not support this requirement because it relies on a general definition of indicators.
II) Model the relation between PPIs.	The <i>Business Intelligence Approach</i> partially supports this requirement because it allows users to model relations between data but not relations between indicators.
III) Consider PPI Variability by restriction.	The <i>Business Intelligence Approach</i> does not support this requirement because it is not possible to model PPIs., therefore it is not possible to model the expected PPI variants a priori.
IV) Consider PPI Variability by extension.	The <i>Business Intelligence Approach</i> does not support this requirement because it is not possible to model PPIs, therefore it is not possible to model the non-expected PPI variants a priori.
V) Model PPIs independently of the language used to model the BP	The <i>Business Intelligence Approach</i> partially supports this requirement because it relies on data modeling instead of indicator modeling.
VI) Model PPI Variability together with Customizable Process Models	The <i>Business Intelligence Approach</i> does not support this requirement because there is no link between Customizable Process Models and Business Intelligence.
VII) Model the relation PPI-Activities of a business process.	The <i>Business Intelligence Approach</i> does not support this requirement because the modeling of indicators is based only on data instead of being based on the link with the processes

2.2.6 PPINOT

The PPINOT approach (Del-Río-Ortega et al., 2019) proposes a language to define and model PPIs together with business processes. This approach is the closest work to our research, which is why we are going to detail the mechanisms of PPI modeling in PPINOT.

PPINOT enhances the PPI modeling as well as the visual representation of business processes and PPI links by instantiating a pre-defined metamodel. PPINOT also makes it possible to express PPI definitions, which were impossible to model in previous approaches such as BPMN (OMG, 2013) and BPEL (Andrews et al., 2003) as analyzed in (Del-Río-Ortega et al., 2013). (OMG, 2013)(Andrews et al., 2003).

PPINOT is a graphical notation for PPIs definitions designed to be used together with Business Process Models and aims at facilitating and automating PPI management design (Del-Río-Ortega et al., 2016). The PPINOT metamodel provides definitions of PPI Variability:

- A PPI varies depending on whether it is defined for all process variants or not.
- A PPI varies depending on the attributes required to define it, which may change depending on the variant in which it is defined.

This approach connects business process elements and PPIs to measure the business process lifecycle. PPINOT improves the expressiveness of the link between PPIs and business processes. However, this approach does not consider Process Variability even if it was validated in a multiple-case study (Del-Río-Ortega et al., 2019).

PPINOT involves PPI definitions in a metamodel, which identifies the main concepts needed to model a PPI. Its purpose is to identify the way in which PPIs are measured, i.e., how the required information can be obtained from a process (Del-Río-Ortega et al., 2019). It considers a set of questions derived from the Specific, Measurable, Attainable, Realistic, Time-sensitive (SMART) criteria. PPINOT uses the Unified Modeling Language (UML) to formalize the models (Del-Río-Ortega et al., 2013).

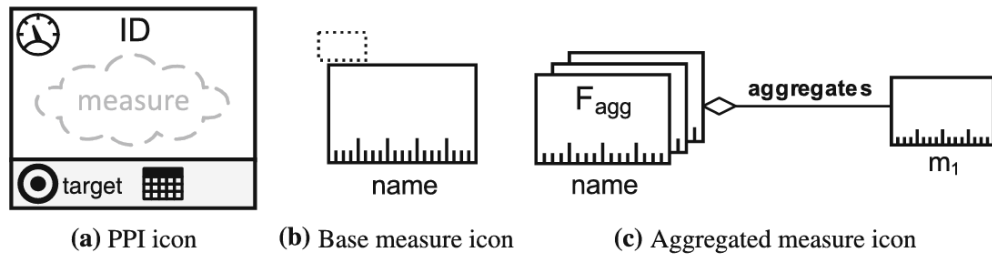


Figure 15 VISUAL PPINOT icons for PPIs, base measures, and aggregated measures (Del-Río-Ortega et al., 2019)

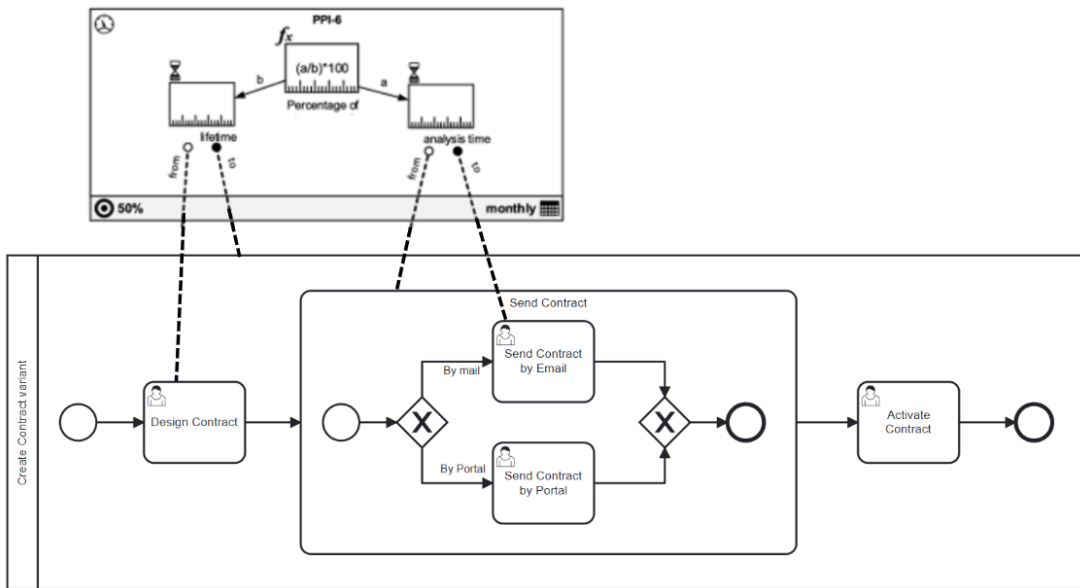


Figure 16 VISUAL PPINOT model example

PPINOT defines PPIs as a rectangle with the measure defining the PPI displayed inside it (Del-Río-Ortega et al., 2019). The PPI value and its scope are displayed at the bottom (cf. Figure 15 (a)). Measures are classified in base measures (cf. Figure 15 (b)), aggregated measures (cf. Figure 15 (c)), and derived measures (cf. Figure 16 (percentage of)). A base measure generates one value for a single-process instance (Del-Río-Ortega et al., 2019). An aggregated measure corresponds to the multi-process instances. PPINOT uses aggregation functions such as: AVG for average, MAX for maximum, MIN for minimum, SUM for summation, etc. (cf. Figure 15 (c)).

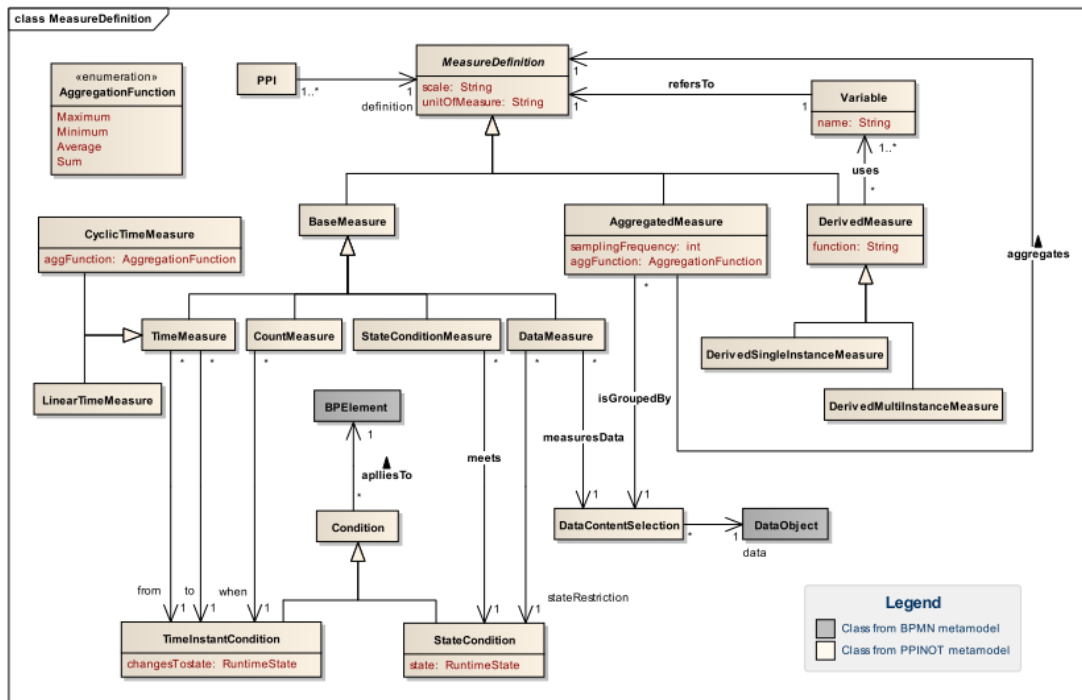


Figure 17 Definition of measures in PPINOT (Del-Río-Ortega et al., 2019)

A more recent version of PPINOT V2 (Estrada-Torres et al., 2021) has implemented PPI variability in Customizable Processes Model. However, the authors used two models, the C-IEPCs (La Rosa et al., 2011) that does not support the process variability extension and the PROVOP (Hallerbach et al., 2010) that does not allow users to avoid behavioral anomalies in particular livelocks and deadlocks when the Customizable Process Model is instantiated. Likewise, PROVOP does not guide users when making customization decisions to select one option or another.

Table 11 presents our requirements to model variable PPIs linked to Customizable Process Models and summary of the evaluated requirements for the PPINOT approach is illustrated on.

Table 11 Requirements analysis of the PPINOT approach

Requirement	PPINOT Approach
I) Define a variable performance indicator.	The <i>PPINOT</i> approach supports this requirement thanks to a formal definition of each PPI. (Del-Río-Ortega et al., 2013) and (Estrada Torres et al., 2016).

II) Model the relation between PPIs.	The <i>PPINOT</i> approach partially supports this requirement because it allows users to model relations between PPIs but only for aggregations.
III) Consider PPI Variability by restriction,	The <i>PPINOT</i> approach supports this requirement because it is possible to select and deploy PPIs according to the business context.
IV) Consider PPI Variability by extension	The <i>PPINOT</i> approach does not support this requirement because it is not possible to model and calculate new PPIs when their variants are not known a priori.
V) Model PPIs independently of the language used to model the BP	The <i>PPINOT</i> approach supports this requirement because on the one hand it allows the modeling of PPIs relying on a language based on measure aggregations. On the other hand, it allows users to model processes using BPMN.
VI) Model PPI Variability together with Customizable Process Models	<p>The <i>PPINOT</i> approach (Del-Río-Ortega et al., 2019) does not support this requirement because it does not consider Customizable Process Models. Therefore, the modeling of PPI variability relies on a non-variable process instead of a Customizable Process Model.</p> <p>The <i>PPINOT V2</i> approach (Estrada-Torres et al., 2021) support this requirement because it consider Customizable Process Models. However, neither of the two models used C-iEPCs (La Rosa et al., 2011) and PROVOP (Hallerbach et al., 2010) do not have an executable and extensible implementation to include the calculation of variable PPIs.</p>
VII) Model the relation PPI-Activities of a business process.	The <i>PPINOT</i> approach supports this requirement because its metamodel considers the link between a PPI and a Business Process Element (cf. Figure 17).

2.2.7 Synthesis of Performance Indicator modeling

Classical approaches of Data Warehouse, Business Intelligence, Business Activity Monitoring (Friedenstab et al., 2012) or Modeling Performance Indicators (Popova & Sharpanskykh, 2010) allow users to model and calculate indicators dealing with the importance of enforcing goals defined by business strategies and metrics. Nevertheless, in the case of Customizable Process Models, the information extracted from business data is insufficient, especially when different PPI definitions depend on flexible evaluation criteria and process variants (cf. Table 12). The PPI Variability allows an advanced definition of variable performance indicator independently of the language used to model the BP (Estrada-Torres et al., 2016). The PPINOT approach allows users to model PPIs together with business processes. However, the PPINOT approach considers neither Customizable Process Models nor the PPI Variability. In summary, the works of (Estrada-Torres et al., 2016) (Del-Río-Ortega et al., 2019) do not allow users to model and define relations between PPI variability and Customizable Process Models.

Table 12 Evaluation of Process Performance Indicators Approaches

Approach	Reference	The expressive capacity of PPIs				The integration level between PPIs and Customizable Process Models		
		I) Define a variable performance indicator.	II) Model the relation between PPIs.	III) Consider PPI Variability by restriction.	IV) Consider PPI Variability by extension.	V) Model PPIs independently of the language used to model the BP.	VI) Model PPIs variability together with Customizable Process Models.	VII) Model the relation PPI-Activities of a business process.
Business Performance Measurement approach	(Kaplan et al., 1992)	No	Yes	No	No	Yes	No	No
Business Process Execution Measurement approach	(Friedenstab et al., 2012)	No	Yes	No	No	Yes	No	Yes
Data warehouse Architecture	(Ngo et al., 2019)	No	No	No	No	Partially	No	No
Business intelligence approach	(Negash & Gray, 2008)	No	Partially	No	No	Partially	No	No
PPINOT	(Del-Río-Ortega et al., 2019)	Yes	Partially	Yes	No	Yes	No	Yes
PPINOT V2	(Estrada-Torres et al., 2021)	Yes	Partially	Yes	No	Yes	Yes	Yes

2.3 SUMMARY AND IMPLICATIONS

As shown in section 2.2, regarding the performance measurement of business processes, research efforts have been made by many approaches that propose languages and architectures to monitor and define PPIs such as (Popova & Sharpanskykh, 2010) or (Saldivar et al., 2016). However, these approaches do not consider neither Customizable Process Models nor the PPI Variability. Other works such as (Friedenstab et al., 2012) have extended the Business Process Model Notation to define business process goals and performance measures, but do not consider any type of variability. It is also worth mentioning that the standard Case Management Model and Notation for decision modelling (OMG, 2016) considers the calculation of the process-related measures but only for non-Customizable Process Models.

From the study of the state of the art, we conclude that BPFM (Business Process Model Families) responds to the modeling requirements when an organization adopts new BP activities. However, BPFM has not been designed to determine the impact of Process Variability in the PPI calculation. Thus, when an organization explores its data sources and uses them as part of a new process, there are no business-related PPI lists. To propose candidate PPIs, managers must rely on their intuition, which causes some PPIs to be redundant, increasing the necessary efforts and resources in its calculation (Rodriguez et al., 2009). Thus, we propose to extend BPFM to guide and facilitate the construction of new PPIs from existing ones. To cover the PPI Variability concern, we were inspired by the PPINOT approach and to cover the Customizable Process Model concern, we relied on the BPFM approach. Our purpose is to model PPI Variability to facilitate PPI definitions (see chapter 3) as well as the integration of PPI modeling into Customizable Process Models relying on BPFM constraints (see chapter 4).

Chapter 3: Modeling Process Performance Indicators Variability integrated to Customizable Process Models: Approach Overview

This chapter presents an overview of our contributions. As explained in the previous chapters, Customizable Process Models and PPIs are generally modeled independently. We propose an integration of these two concepts to provide a method and a tool to calculate new PPIs depending on the variants of a business process, as well as to recalculate an existing PPI depending on the requirements and definitions of stakeholders. To integrate Performance Indicator Variability with Customizable Process Models, we first describe the main concepts of the Process Performance Indicator Calculation Tree (PPICT) thoroughly described in section 3.1.1, which allows the modeling of the PPI variability linked to Customizable Process Models based on the Business Process Feature Model (BPFM) approach. Then, the Process Performance Indicator Calculation (PPIC) method is proposed, which extends the BPFM method, previously presented in the state of the art (cf. Figure 13). This is to guarantee an adequate design of PPIs linked to a Customizable Process Model under a framework relying on feature models.

Our contribution is based on three fundamental aspects:

- I) The PPICT, a model to represent the variability of PPIs, i.e., a family of PPIs (overview in section 3.1.1 and detailed in chapter 4).
- II) The PPIC method, which enables and guides the modeling and the analysis of the PPI variability linked to Customizable Process Models (overview in section 3.1.2, detailed in chapter 5 and validated in chapter 6).
- III) The PPIC Tool, which supports the PPIC Method (overview in the section 3.1.3, detailed in chapter 7).

These three main bricks of our proposal are briefly described in the first part of this chapter and detailed in the further ones. The second part of this chapter presents the human centered research methodology that has been put in place to build this work and the experiments.

3.1 OVERVIEW OF THE THREE MAIN PROPOSITIONS

3.1.1 THE PPICT OVERVIEW

Definition: *Process Performance Indicator Calculation Tree (PPICT)* (cf. Figure 18) arises from the need to model and calculate the PPI Variability linked to Customizable Process Models according to process variants and deployment conditions. The PPICT proposes to model PPIs in the form of a tree where each leaf corresponds to a PPI, and each branch corresponds to the link between PPIs. The PPICT also proposes relationships between the activities of a Customizable Process and one or more PPIs. The links between PPIs are defined as a parent-child relationship in the tree. We call this link the relationship between a reference PPI and a variant PPI (cf. Figure 18). The addition of a new PPI can imply the inclusion of execution of another PPI (cf. Figure 18).

The PPICT is formalized through a metamodel (abstract syntax) and represented with a graphical notation (concrete syntax) which will be presented in chapter 4.

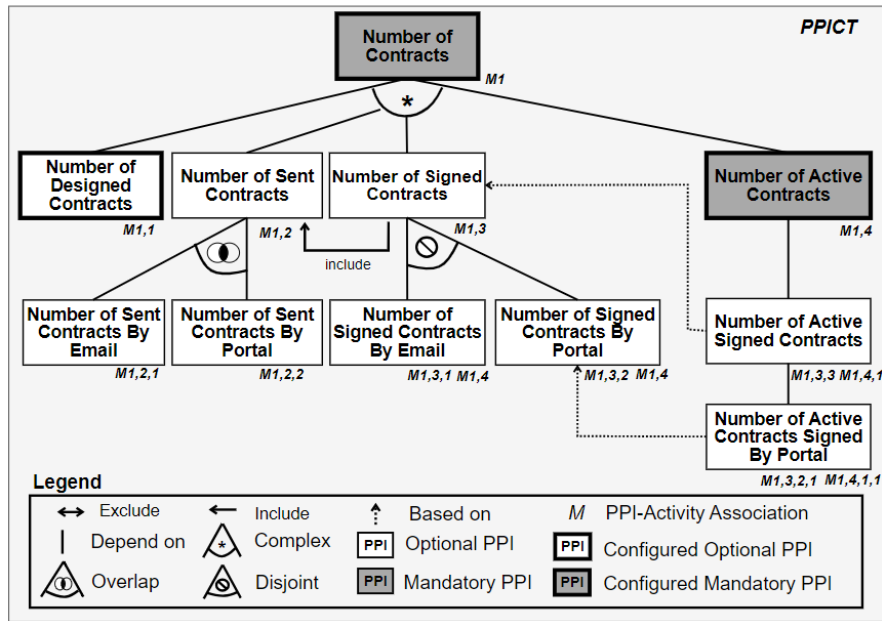


Figure 18 PPICT example

Modeling the PPIs as leaves of a tree allows a visible hierarchy between indicators, which makes it possible to generalize the PPI root calculation and detail the calculation of the PPI variants. When a new PPI level is added to the tree, it is necessary to specify the calculation of the new PPI. The PPICT uses the set theory, therefore all tuples of a variant PPI before applying any aggregation or any group by are subsets smaller or equal to all tuples of its reference PPI before applying any aggregation or any group by. For this reason, the PPI variant is totally dependent on its reference. A PPI variant has only one parent. However, the PPICT allows the calculation of a PPI variant depending on more than one reference PPI.

3.1.2 OVERVIEW OF THE PPIC METHOD

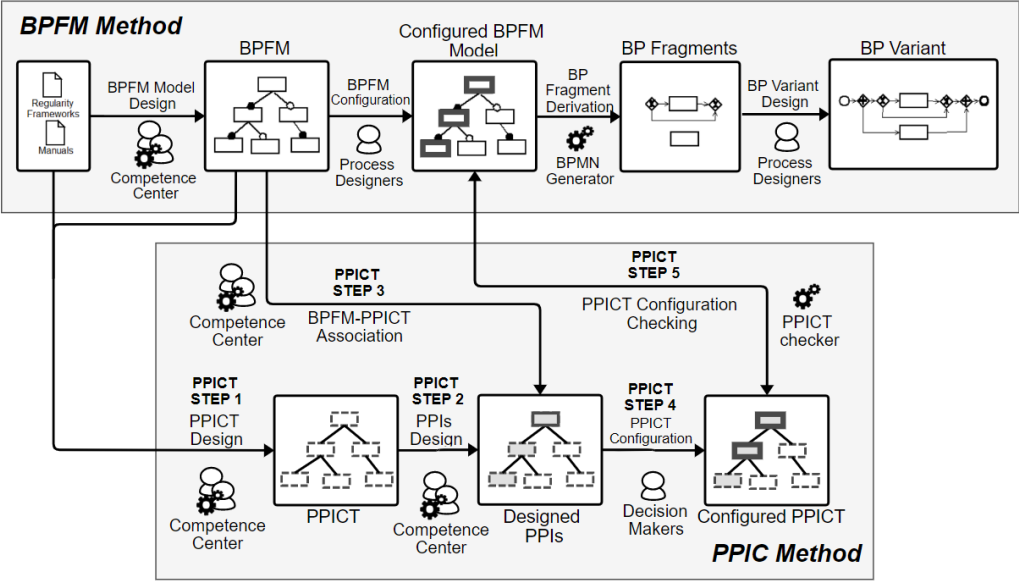
The PPIC method describes the process that must be followed to design, model and calculate the PPIs linked to a Customizable Process Model that exists as an input of the PPIC Method. This method allows the support of PPI Variability modeling through five design stages, which are defined below and detailed in chapter 5.

- 1) **The PPICT design:** this stage refers to the manual addition of all PPI family members using the PPICT graphical notation. The goal is to build the skeleton of the PPI tree for a given PPI family.

- II) **The PPI design:** this stage specifies that all PPI family members must be designed according to stakeholders' definitions and design criteria. The goal of this stage is to decide if the PPI should be mandatory or could be optional, define the PPI representation (value representation, listing representation, geographical representation or chart representation) as well as to indicate the measure type.
- III) **The BPFM and PPICT association:** this stage refers to the association between PPIs and the process activities of the existing Customizable Process Model. The goal is to map PPI-Activity according to the Customizable Process Model performance requirements with a process modeled using the BPFM method.
- IV) **The PPICT configuration:** this stage defines the configuration process of PPIs family members. The configuration of a PPI depends on its status (mandatory or optional) and on stakeholders' specifications. The goal of this stage is to define which PPI family members must be considered for the customized process model, considering business regulations and decision makers' criteria.
- V) **The PPICT-BPFM configuration checking:** this stage focuses on the alignment between configured PPIs and configured activities. The goal is to align the PPICT configuration with the BPFM configuration, i.e., check if the PPICT configuration matches with the BPFM configuration. In other words, this step checks which members of the configured PPI family do not match with the BP configuration.

The PPIC method extends the BPFM method (Cognini et al., 2016). The PPIC method does not exist as a separate entity. It necessarily requires the modeling of a Customizable Process Model before step one of the PPIC Method. The PPICT can be exploited independently since it allows the modeling of the PPI Variability without considering the Customizable Process. The PPIC method requires different stakeholders. For instance, for step 1, 2 and 3 a competence center is needed, which is composed of process designers, PPI designers and PPI developers and decision-makers. For step 4 only decision-makers are needed. Step 5 can automatically check the PPI and process configuration (in our case, through a tool). The goal is to create a

common framework between a Customizable Process Model, PPI Variability, and business evaluation strategies (cf. Figure 19).



3.1.3 OVERVIEW OF PPIC PROTOTYPE

The proposed prototype is based on a modeling tool called ADOxx⁷. The goal of this tool is to implement the PPIC method based on the PPICT metamodel, allowing the user to model a tree of PPIs linked to a BPFM model. The development was inspired by the implementation of the tool supporting the BPFM method, which allows developers and users to manipulate the concepts of the metamodel. The prototype allows the modeling of PPI Variability but also Customizable Process Models, thus enabling the alignment check between the PPI configuration and the displayed process activities. The prototype is detailed in chapter 7.

The prototype is divided into two views. The first view is the individual modeling of PPIs without any link to a Customizable Process Model. The second view is the modeling of the PPI Variability linked to a Customizable Process Model. Each PPI has a certain number of attributes to be defined. A PPI can be either a PPI reference or a PPI variant. These attributes are presented as a form. Therefore, the user can select or complete the characteristics, the calculation method and deployment conditions of

Figure 19 Steps of PPIC Method

⁷ www.adoxx.org/live/home

a PPI. This prototype is available on the OMLAB⁸ page as a modeling tool for variable PPIs linked to Customizable Process Models.

3.2 RESEARCH DESIGN RELYING ON THE THEDRE METHOD

In this subsection we describe the research design and outline adopted to design and develop our contributions and our proposed tool using user-centered experiments. Our research relies on the THEDRE method (Traceable Human Experimental Design Research) (Mandran & Dupuy-Chessa, 2018). The purpose of THEDRE is to define a method to conduct Human-Centered Computer Science Research. The goal is to involve humans in building and evaluating research contributions. THEDRE also aims to improve exchanges between researchers and specialists in production and data analysis to identify the limits of the research and to organize its experimental phases.

Concretely, the THEDRE method offers:

- I) A global method for conducting research, divided into subprocesses and blocks to better trace the research and its result, e.g., experimental material.
- II) A set of tasks and targets to check when seeking to assess the relevance of research, e.g., check essential material for research or experiments.
- III) Example guides and management tools to conduct research and result evaluations, e.g., interview guides.

3.2.1 Methodology

We rely on the THEDRE method since it allows us to lead user-centered experimental methods (Mandran & Dupuy-Chessa, 2018). This method is a formalized process to manage research with quality management tools. It allows its actors, e.g., researchers, developers and methodologists, to be guided at each step of a research process. THEDRE has been structured according to the **Plan Do Check Act (PDCA)** continuous quality improvement cycle. **Plan** for the research plan, **Do** for the development of the experimental material and execution of experiments, **Check** for the evaluation of the experiment results and **Act** for the communication of experiment

⁸ <https://www.omilab.org/>

results and the validation of the proposition through research papers. After an Act phase, we can start a new cycle or finish the research with communication. Below, the four phases are detailed regarding our process:

PLAN: in this stage, we focused on research construction, and we set the following three research targets to achieve concerning developments, experiments, and communications.

- I) How to integrate the PPIs into the Customizable Process Models?
- II) How to model and calculate the variable PPIs into the Customizable Process Models?
- III) How to recalculate existing PPIs or calculate new ones according to the Customized Process Model?

We have set the following experimental targets: I) involve indicator developers in the modelling of PPIs according to Customizable Process Models, II) explore how users express their PPI definitions, and III) identify users' modeling methodologies and calculation practices of PPIs.

DO (EXPERIMENT): in this stage we focused on the development of experiments to assess the main concepts of our research instrument. Five experiments with experts and five experiments with novices were carried out to validate the proposed model concerning the creation of a modeling language for process performance indicators. During this qualitative experiment, there were intercalated practical work cases and discussion phases supported by interview guides. Thus, experts and novices expressed their views individually on both improvements and questions to know how clear the proposed concepts are and to know the point of view of each participant about the contribution. We used independent, dependent, and classificatory variables to define the research hypotheses. Thus, we caught the approach taken by users and briefly discussed the data gathering procedures that were used through observations, questionnaires, and interviews.

CHECK (CONTROL): at this stage we assessed the experimental results and controlling targets. Some research targets were checked such as methodological hypotheses and clarity of concepts. This step was crucial to design the experiments and control the Progress of the research.

ACT (BUILDING AND DECISION): during this stage, we obtained some scientific knowledge from the experimental results and their limits. The results were documented to communicate our progress and observe possible improvements to our model and method to develop a software tool that would model PPI Variability linked to Business Process Families.

COMMUNICATION: publication of results linked to the research instrument, according to our real case study. This sub-process is the last step before iterating on a new research question.

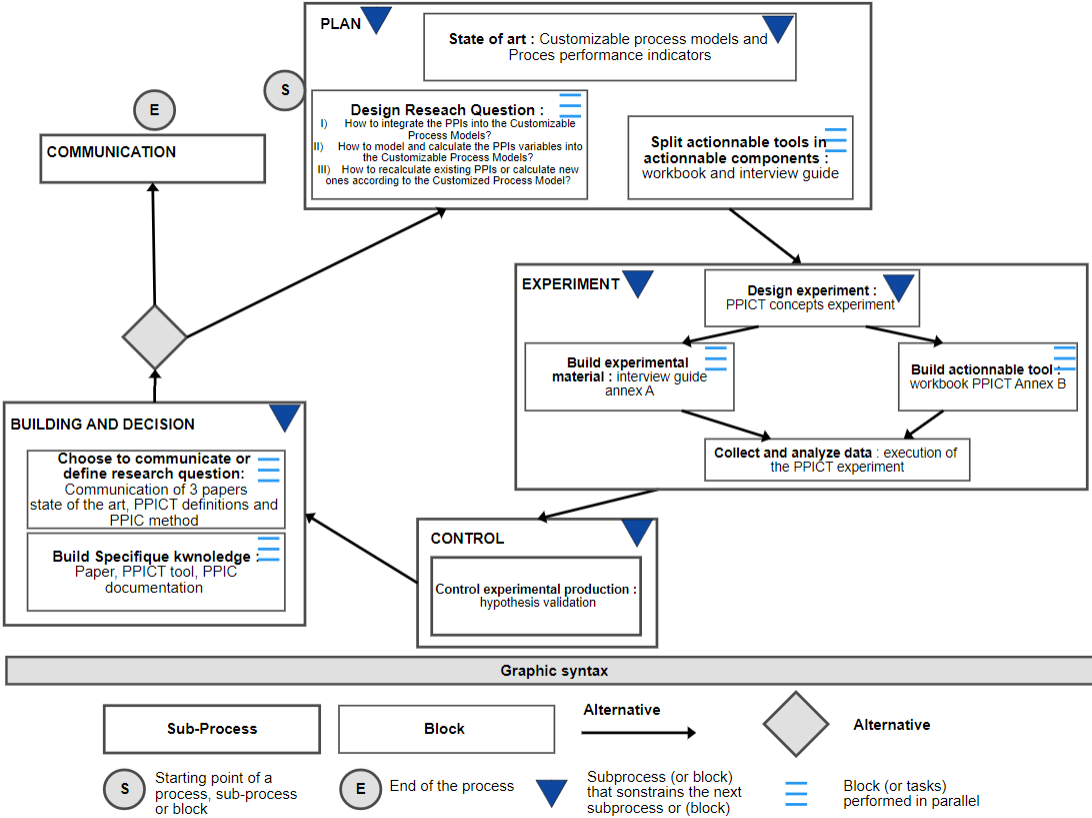


Figure 20 Ph.D. process research based on The THEDRE Process (Mandran & Dupuy-Chessa, 2018)

In the next chapter we will explain in more details PPICT, the model proposed to support and represent the PPI Variability linked to Customizable Process Models.

Chapter 4: Process Performance Indicator Calculation Tree (PPICT)

This chapter presents our first contribution: the Process Performance Indicator Calculation Tree PPICT. We define here the main concepts of our model: Process Performance Indicator PPI, reference PPI, variant PPI, *family of PPI*, relations between PPIs, relations between PPIs and activities and rules concerning PPI variants and PPI references. These rules are represented using relational algebra operators.

4.1 DEFINITIONS

Definition: Family of PPIs. We define a *family of PPIs* as a set of PPIs of the same process family, these PPIs share common operators such as projections, selections and joins. Likewise, a *family of PPIs* is defined as the calculation variability of a set of PPIs.

We use a *Query Tree* in relational algebra to illustrate a PPI. So that a *family of PPIs* is the set of several *Query Trees* (cf. Figure 21). Each node of the indicator tree represents a query operator: projection π , aggregation Υ , selection σ , cartesian product \times , Union \cup , Difference \setminus , Intersection \cap , Join \bowtie and Relational division \div . Leaves correspond to relations/tables. Due to these shared similarities, the development of a family of PPIs improves the reuse and profitability of decision reports corresponding to the Customized Process Models. A family of PPIs must be specified by a competence center involving PPI designers, PPI developers and decision-makers. Additionally, a family of PPIs is linked to a family of processes to measure and evaluate the customized process model. For example, The *Number of Contracts (NC) PPI*, which is illustrated in the NC tree of Figure 21 whose nodes are operators and arcs represent the tuple stream

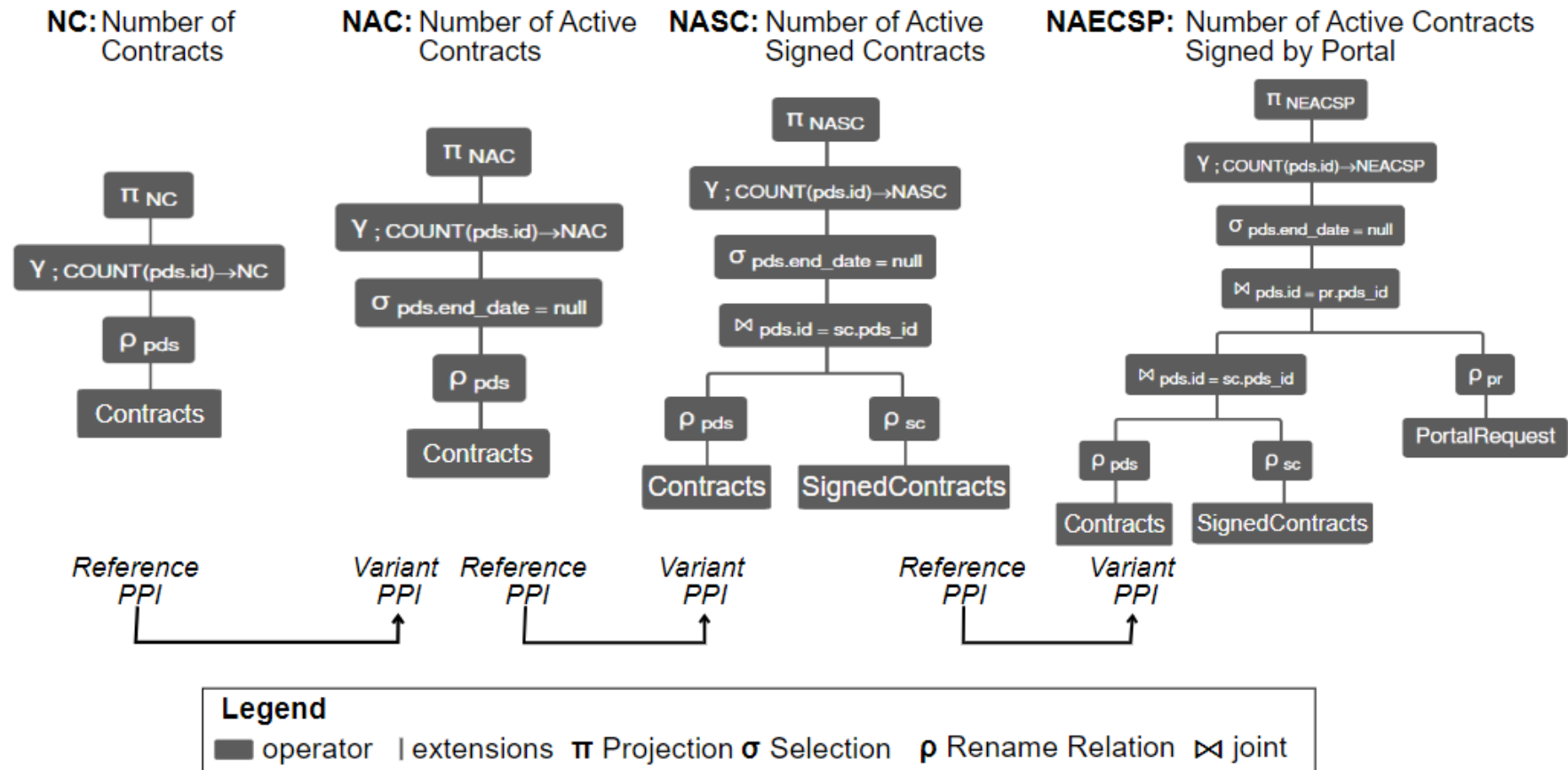


Figure 21 Family of PPIs: PPI Query and relationship

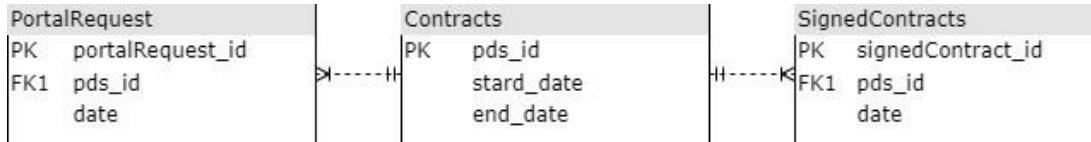


Figure 22 Database model of the family of PPIs

Below 3 examples of PPI expressed in SQL are illustrated.

For the PPI NC *Number of Contracts*, the associated query is:

```

SELECT count(pds.id)
FROM contrats psd;
  
```

For the PPI NAC *Number of Active Contracts*, the associated query is:

```

SELECT count(pds.id)
FROM contrats psd
WHERE pds.enddate IS NULL;
  
```

For the PPI NASC *Number of Active Signed Contracts*, the associated query is:

```

SELECT count(pds.id) FROM contrats psd
INNER JOIN scignedcontrats spds ON spds.id
    = pds.id
WHERE pds.enddate IS NULL;
  
```

Short Definition: PPICT. We define a Process Performance Indicator Calculation Tree (PPICT) as a tree of PPIs that captures how PPIs can be calculated, modeled, linked and reused in the context of a Customizable Process Model. A PPICT is a model of PPIs organized as a tree with just one PPI root and several PPI leaves.

To model a family of PPIs, a PPICT provides a global representation of PPI calculation through the systematic modeling of variability and common points of PPIs. Indeed, the PPICT defines the available PPI members of a family of PPIs and dependencies between them. The PPICT approach aims at combining BP family and PPI family modeling relying on the aforementioned Business Process Feature Model (BPFM). Below, we propose definitions of modeling PPI families represented in the PPICT approach. We extend the PPI definitions proposed by (Estrada-Torres et al., 2016) including relations between PPIs and process activities.

A PPI is a quantifiable metric focused on evaluating the performance of a business process in terms of effectiveness and efficiency (Estrada-Torres et al., 2016). PPIs are directly measured by data generated within the process flow and are used for process controlling and continuous optimization (Del-Río-Ortega et al., 2013). A PPI is calculated by using a set of tuples represented either as a value, a percentage, a list, a map or a chart.

A *family of PPIs* allows the management and evaluation of the process performance. It may be computed regarding a calculation, i.e., the result of the computation of a PPI, which is referred to an *aggregation* or a *group by* of tuples of a PPI. A PPI can be calculated differently depending on:

- I) the desired process model customization, i.e., the process variant customized for a particular context
- II) the definition of metrics by stakeholders, i.e., the stakeholder's requirements for process analysis, e.g., having different definitions of an e-contract (contract signed by mail only, contract sent and signed by a web portal)
- III) the activities involved, i.e., the activities used to calculate a PPI. For example, to calculate the PPI *Number of Sent and Signed Contracts*, it is necessary to use the data that is generated by the *send contract* activity and the *sign contract* activity.

Definition: Reference PPI. We define a *Reference PPI* as a PPI that serves as the basis for calculating its variant PPIs (cf. Figure 23) denoted as *R. PPI* (cf. Figure 24). Figure 21 shows an example where the *Number of Contracts* NC is the reference PPI of the *Number of Active Contracts* NAC PPI. In this example, the operator $\sigma_{pds.end_date = null}$ is added into the original query of NC to extract only Activated Contracts. It can also be seen that NAC is the reference PPI of NASC. In which the added the operator $\bowtie_{pds.id = sc.pds_id}$ and ρ_{SC} extract only the active signed contracts. This means that every variant PPI only adds operators to the original query of its reference PPI. i.e., a variant PPI does not delete any operators from its reference PPI.

The operators that a reference PPI can use are projection, aggregation (count(), max(), min(), sum(), average()), selection, renaming, intersection, difference and join. For instance, the reference PPI *Number of Contracts* has:

- the projection π_{NC}
- the aggregation $\Upsilon; \text{count}(pds.id) \rightarrow NC$
- the relation $\rho_{pds} \text{Contracts}$
- the renaming $NC \text{ and } pds$.

Any of these operators can be added in a variant PPI if all tuples of a reference PPI contains all the possible tuples of this variant PPIs. All tuples of variant PPIs must be a subset of the PPI before applying any aggregation or any *Group By* are subsets smaller or equal to all tuples of its reference PPI before applying any aggregation or any *Group By* (cf. Figure 21).

Definition: Variant PPI. We define a *Variant PPI* as a PPI derived from its reference PPI (cf. Figure 23) denoted as $V.PPI$ (cf. Figure 24). For this reason, all tuples of a variant PPI before applying any *aggregation* or any *group by* are subsets smaller or equal to all tuples of its reference PPI before applying any *aggregation* or any *group by*. This inclusion rule is called μ , i.e., $\mu(V.PPI) \subseteq \mu(R.PPI)$. This means that a variant PPI must only have one reference PPI that meets this condition. A variant PPI contains at least:

- The same projections as its reference, e.g., π_{NC}
- The aggregation $\Upsilon; \text{count}(pds.id) \rightarrow NC$
- The same joins as its reference, e.g., $\bowtie_{pds.id=sc.pds_id}$
- The same relations as its reference, e.g., $\rho_{pds} \text{Contracts}$
- The same selections as its reference. e.g., $\sigma_{pds.end_date = null}$

A variant PPI cannot remove any operators from the reference PPI. A variant PPI can have a different graphical representation than its reference PPI and its siblings derived from the same reference. A new representation of the reference PPI implies a new variant PPI. For example, if the reference PPI representation is a value and a percentage needed, a new variant PPI must be calculated.

Extended Definition: PPICT. We define a PPICT also as a set of PPIs organized in a tree, where the tree's root identifies a family of PPIs (cf. Figure 23, PPI₁). Each PPI of the internal tree structure is a PPI reference of another PPI, i.e., each PPI that is not a tree leaf is a reference PPI including the root (cf. Figure 23, PPI₁ ; PPI_{1,2}).

Regarding PPI leaves, they are variants of a PPI reference (cf. Figure 23, PPI_{1.1} ; PPI_{1.2.1} ; PPI_{1.3.1} ; PPI_{1.3}). Additionally, 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 PPI leaves with variant roles and the PPI-root with the reference role (cf. Figure 23). Additionally, the PPICT manage different calculation levels such as PPI specifications, connections and constraints between reference PPI and variant PPIs (*Depend on constraint* and *Based on constraint*). These constraints are detailed in the next section as well as the constraints between PPIs and process' activities.

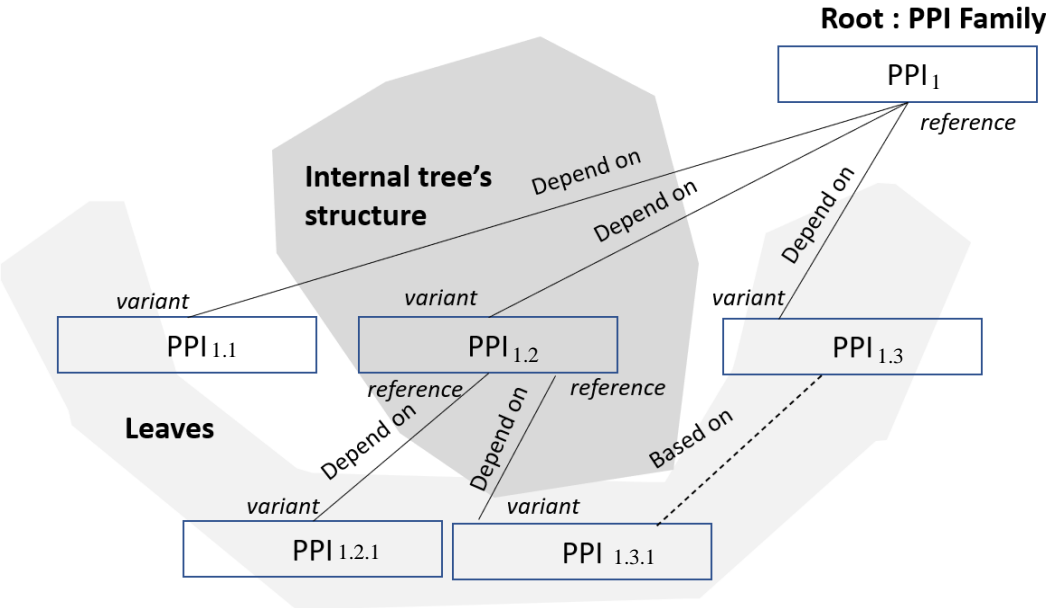


Figure 23 Process Performance Indicator Calculation tree (internal tree's structure)

4.2 PPICT CONSTRAINTS

The rules that guide users to link the BPFM and the PPI Tree are presented here. The PPICT's constraints are divided into 3 groups: Binary, Multiple and PPI-Activity constraints. These constraints represent both dependencies between PPIs and associations between PPIs-activities. Every individual variant PPI added to the tree must be connected to its reference PPI in a binary relationship. A PPI can be the reference of several individual variants, but a variant PPI depends on only one reference.

PPICT Binary Constraints are defined as follows:

- **Definition: Depend-on.** We define a Depend-on constraint as a constraint that applies the rule $\mu(V.PPI) \subseteq \mu(R.PPI)$ which specifies that all tuples of a variant PPI before applying any *aggregation* or any *group by* are subsets smaller or equal to all tuples of its reference PPI before applying any *aggregation* or any *group by*. In Figure 24, Binary constraints show how the PPIs are related using the constraint Depend-on. An example of this constraint is the relation between *Number of Contracts* and the *Number of Active Contracts* of the Figure 21. This constraint is mandatory to add PPI variants.
- **Definition: Based-on.** We define a Based-on constraint as a constraint that specifies that a variant PPI could be based on one or more reference PPIs, (cf. Figure 24 Binary constraints). This constraint is optional. When there are two or more possible *reference PPIs*, one is chosen, which becomes the *reference PPI* using the constraint *Depend on* and the others become *based-on*.

Every group of variant PPIs added to the tree must be connected to its reference PPI through a multiple relationship. For this, we propose PPICT Constraints, detailed below, which specify the dependency between a PPI and a group of PPIs to determine their role in the tree. A PPI can be the reference of a group of variants, but each variant must have only one reference.

PPICT multiple Constraints are defined as follows:

- **Definition: Overlap Constraint.** We define an Overlap Constraint as a constraint that specifies the intersection Overlap between variant PPIs of the same reference PPI. Thus, all resulting tuples of this intersection between variant PPIs, before applying any *aggregation* or any *group by*, are subsets smaller or equal to all tuples of its reference PPI, before applying any *aggregation* or any *group by*, i.e., $\mu(V.PPI_i) \cap \mu(V.PPI_j) \subseteq \mu(R.PPI)$ where $\mu(V.PPI_i) \subseteq \mu(R.PPI)$ and $\mu(V.PPI_j) \subseteq \mu(R.PPI)$ (cf. Figure 24).
- **Definition: Disjoint Constraint.** We define a Disjoint Constraint as a constraint that specifies the intersection Disjoint between variant PPIs of the same reference PPI. Thus, all resulting tuples of this intersection between

variant PPIs, before applying any *aggregation* or any *group by*, are equal to zero \emptyset , i.e., $\mu(V.PPI_i) \cap \mu(R.PPI_j) = \emptyset$, where $\mu(V.PPI_i) \subseteq \mu(R.PPI)$ and $\mu(V.PPI_j) \subseteq \mu(R.PPI)$. Thus, all intersections between variant PPIs are disjoint sets (cf. Figure 24).

- **Definition: Complex Constraint.** We define a Complex constraint as a constraint that specifies that the intersection between variant PPIs of the same reference PPI is Complex. Thus, all resulting tuples of this intersection between variant PPIs, before applying any *aggregation* or any *group by*, can be subsets smaller or equal to all tuples of its reference PPI, before applying any *aggregation* or any *group by* or can be equal to zero \emptyset , i.e., intersections between variant PPIs can be overlapping or disjoint sets.

	<i>PPICT Constraints</i>	<i>Mandatory PPICT Rules</i>
<i>Binary Constraints</i>		$\mu(V.PPI_i) \subseteq \mu(R.PPI)$ <i>Where V.PPIi is a variant PPI of R.PPI</i>
		$\mu(V.PPI_i) \subseteq \mu(R.PPI)$ <i>Where V.PPIi is a variant PPI of R.PPI</i>
<i>Multiple Constraints</i>	<p><i>Overlap Constraint</i></p>	$\mu(V.PPI_i) \subseteq \mu(R.PPI)$ $\mu(V.PPI_j) \subseteq \mu(R.PPI)$ $\mu(VV.PPI_i) \cap \mu(VV.PPI_j) \subseteq \mu(VR.PPI)$ <i>Where V.PPIi and V.PPIj are variant PPIs of R.PPI</i>
	<p><i>Disjoint Constraint</i></p>	$\mu(V.PPI_i) \subseteq \mu(R.PPI)$ $\mu(V.PPI_j) \subseteq \mu(R.PPI)$ $\mu(VV.PPI_i) \cap \mu(VV.PPI_j) = \emptyset$ <i>Where V.PPIi and V.PPIj are variant PPIs of R.PPI</i>
	<p><i>Complex Constraint</i></p>	$\mu(V.PPI_i) \subseteq \mu(R.PPI)$ $\mu(V.PPI_j) \subseteq \mu(R.PPI)$ $\mu(VV.PPI_i) \cap \mu(VV.PPI_j) = \emptyset$ <i>Where V.PPIi and V.PPIj are variant PPIs of R.PPI</i>
	<p>Legend <i>R.PPI = Reference PPI V.PPI = Variant PPI</i></p>	

Figure 24 PPICT Constraints and Rules: each constraint respects one or more modeling rules

4.3 PPICT-ACTIVITY CONSTRAINT

Since the PPIC method is an extension of the BPFM method, it is necessary to link the PPI family modeling with the BP family modeling. In this sense, we propose the PPI-Activity Constraint, 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:

- Definition: PPI-Activity Constraint.** We define a PPI-Activity Constraint as the mapping of a PPI with at least one activity of the BP family, i.e., BPFM Model (cf. Figure 25). This mapping M seeks to identify the activity or activities to be evaluated with an indicator. For example, in Figure 25 M1 links the reference PPI R.PPI and Activity A.

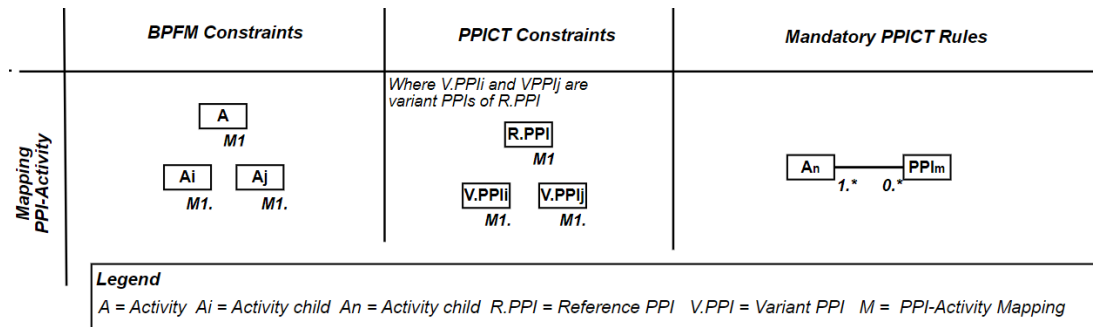


Figure 25 PPICT-BPFM CONSTRAINT

Essentially, PPICT constraints associate PPIs to the BPFM model, and design PPI family models. The next section presents the PPICT Metamodel, which formalizes the definition of the constraints and the links with PPIs.

4.4 THE PPICT METAMODEL

This section presents the PPICT metamodel, which supports the PPI Variability modeling. The PPICT metamodel is represented in UML, it describes the concepts that have been described before without the mapping which will be detailed in the next chapter. The PPICT metamodel uses the notion of PPI as a central point of calculation conditions and constraints.

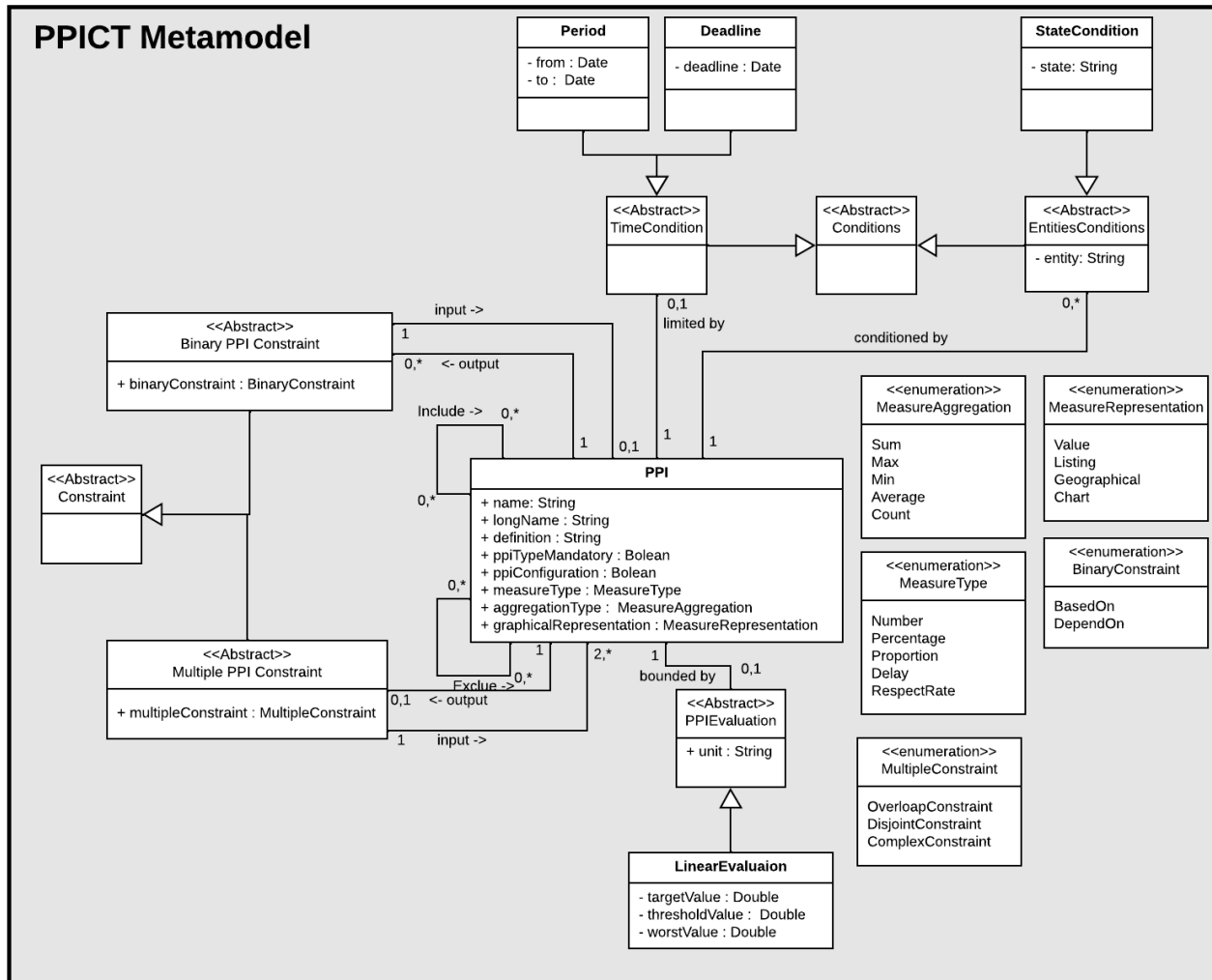


Figure 26 Metamodel of Process Performance Indicator Calculation Tree PPICT

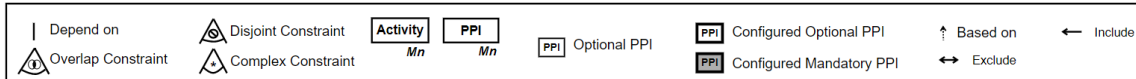


Figure 27 PPICT Concrete Syntax

4.4.1 PPICT Metamodel Classes

PPI class: represents the process performance indicator and the necessary information to calculate it. Every PPI must be easily identified by stakeholders. Therefore, 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*, cf. Figure 26.

Conditions Class: a condition represents dependencies of a PPI regarding a particular value of a date or entity. Every PPI added to the tree could have conditions regarding the value of an entity, i.e., the value of an attribute of a table.

Entities Conditions: we use the Entities Conditions to define a filter to apply in a PPI, i.e., selection (σ operator).

- **State Condition** is a sub class of **Entities Conditions** which specifies a particular filter to be applied to an entity, i.e., applying a filter with a particular value of an entity. An example of this condition can be when we are looking for *the Number of Send Contracts in Paris*. In this case the entity is the city, and its value is Paris.

Every group of variant PPIs added to the tree could have **Time Condition** regarding the space of time Period and Deadline. These conditions depend directly on the result that is sought from a PPI in terms of temporal duration and time limits:

- **Period** is a sub class of **Time Condition** which specifies the filter to apply regarding the space of time that the PPI considers. This condition uses the duration between FROM and TO of a SQL language. An example of this condition could be when the distributor wants to know the *Number of Sent Contracts by Email between 01/01/2022 to 12/31/2022*.
- **Deadline** is a sub class of **Time Condition** which specifies the filter to apply regarding a limit date that should not be exceeded. For instance, a

deadline condition could be designed at 12/31/2022 Hence, for the PPIs *Number of Sent Contracts before the 12/31/2022*, what is sought are all contracts signed before this deadline regardless of the start date.

Constraint: as mentioned before, the PPICT constraints are divided into 3 groups: binary, multiple and PPI-Activity, which are detailed below. These Constraints represent both dependencies between reference PPIs and variant PPIs and between PPI and Activities . An input and an output relation are represented in the metamodel to be implemented on the metamodeling platform ADOxx.

The **Binary PPI Constraints class** specify the relation between PPIs when each individual variant PPI that is added to the tree must be connected to its reference PPI using a **Binary Constraint Enumeration**:

- A **Depend-on constraint** specifies that all tuples of a variant PPI before applying any *aggregation* or any *group by* are subsets smaller or equal to all tuples of **its reference PPI** before applying any *aggregation* or any *group by*. 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 part of the PPI reference *Number of Active Contracts*.
- A **Based-on constraint** specifies that all tuples of a variant PPI before applying any *aggregation* or any *group by* are subsets smaller or equal to all tuples of **various reference PPI** before applying any *aggregation* or any *group by*. For example, a contract that was signed and activated this week will be part of the PPI *Number of Active Contracts this week* and part of the PPI *Number of Signed Contracts this week*. That is why the variant PPI *Number of Signed and Active Contracts this week* could potentially have 2 reference PPIs. In this case the variant PPI must have just one reference PPI but it is also based on one reference PPI more.

The **Multiple PPI Constraints** specify the dependency between a reference PPI and a group of PPI variants using a **Multiple Constraint Enumeration**:

- An **Overlap constraint** specifies that all tuples of a variant PPI before applying any *aggregation* or any *group by* are subsets smaller or equal to all tuples of its reference PPI before applying any *aggregation* or any *group by*, i.e., all intersections between variant PPIs are overlap sets. An

example of this constraint can be when a contract is sent by email and by a web portal (cf. Figure 30). This contract will be part of the PPI *Number of Contracts Sent by Email* and part of the PPI *Number of Contracts Sent by Portal*.

- A **Disjoint constraint** specifies that all intersections between PPI variants are equal to zero \emptyset , i.e., all intersections between variant PPIs are disjoint sets. An example of this constraint can be when a user signs a contract. It could be done either by email **or** by a web portal but not by both platforms, cf. Figure 30. Hence, the signed contract will be part of the PPI *Number of Sent Contracts by Email* **or** part of the PPI *Number of Sent Contracts by Portal*.
- A **Complex constraint** specifies that all intersections between variant PPIs can be equal to zero \emptyset or not, i.e., intersections between variant PPIs can be disjoint sets or not. For instance, a contract can be designed and sent but not yet activated, cf. Figure 30. Hence, this contract will be part of the PPIs *Number of Designed Contracts* and *Number of Sent Contracts* but will not be part of the PPI *Number of Active Contracts*.

According to the link between a PPI with one or more activities of the process model it is necessary to propose a class that manage this relation :

- PPI-Activiti Constraint specifies that an activity can have zero or several associated PPIs and that a PPI must have at least one associated activity to be calculated:

Regarding the evaluation of an instance of a PPI we propose an abstract class called **PPI Evaluation** with just the attribute of *unit*. This unit depends on what the user wants to evaluate.

- **Linear Evolution** is a sub class of **PPI Evaluation** which specifies the minimum, average and maximum possible value of an instance of a PPI. These values are chosen carefully by decision-makers who evaluate the results of each instance and a set of instances.

4.4.2 PPI Class Attributes

A PPI has an attribute **Measure Type** that determines how it is calculated depending on the result that the client is looking for. We define each measure type as follows:

- **Measure Type:**
 - **Number** specifies the PPI calculation according to the number of tuples, before applying any *aggregation* or any *group by*, that validate a predicate, e.g., *83 Contracts are Active*.
 - **Percentage** specifies the PPI calculation according to the percentage of tuples, before applying any *aggregation* or any *group by*, that validate a predicate, e.g., *60% of Contracts are Active*.
 - **Proportion** specifies the PPI calculation according to the proportion between tuples, before applying any *aggregation* or any *group by*, and a target value, e.g., *(3/5) 3 out of 5 Contracts are Active*.
 - **Delay** specifies the PPI calculation according to the difference between creation dates of tuples, before applying any *aggregation* or any *group by*, e.g., *3 Delay Days in Activating Contracts (Datetoday – Datedeadline)*.
 - **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 (Datetoday – Datedeadline)/Value*.

Furthermore, every PPI has an attribute **Graphical Representation** to visualize tuples in different ways, depending on the type of information that the decision-maker wants to analyze, cf. Figure 26. We define each measure representation as follows:

- **Graphical Representation:**
 - **Value** representation is a type of representation that allows the visualization of a set of tuples as a value, e.g., *83, 60%, 5/3, 3 or 3/2* after applying an *aggregate* or a *group by*.
 - **Listing** representation is a type of representation that allows the visualization of a set of tuples as a listing. It requires additional projections to analyze complementary information linked to tuples, e.g.,

Contract Creation Date, Contract Activation Date, Contract holder, Holder's phone, among others.

- **Geographical** representation is a type of representation that makes it possible to geographically visualize a set of tuples. It requires grouping the geographical tuples, e.g., *by City*.
- **Chart** representation is a type of representation that allows the visualization of a set of tuples as a Chart. It requires grouping tuples regardless of their type, e.g., *by Year, by Type of public service*, among others.

Likewise, every PPI has the attribute **Aggregation Type** to aggregate tuples in diverse ways. It depends on the type of performance indicator that the decision maker wants to analyze, cf. Figure 26.

- **Aggregation Type:**

- **Sum:** the SUM() aggregation function is the adding of positive values and subtracting of absolute values from negative values.
- **Count:** the COUNT() aggregation function is used to count the number of tuples in a table. Knowing the number of rows of a table is especially useful in many cases, for example knowing how many users are in a table.
- **Avg:** the AVG() aggregation function is used to calculate an average value over a set of numeric and non-null type tuples.
- **Min:** the MIN() aggregate function returns the smallest value from a selected column. This function applies to both numeric and alphanumeric data.
- **Max:** the MAX() aggregation function returns the maximum value of a column in a record set. The function can be applied to both numeric and alphanumeric data.

Additionally, a PPI can be optional or not, according to the business context or decision makers requirements, for instance, we must include the PPI *Number of Signed Contracts* or not.

- **PPI Type Mandatory:**

- A **PPI Type** specifies if the PPI is optional or mandatory. This attribute is a Boolean, whose value is true if the PPI is mandatory (cf. Figure 27 for concrete syntax and Figure 26 for abstract syntax).

A PPI can be configured optionally or not, according to the business context or decision-makers' requirements, for instance, we can configure or not the PPI *Number of Signed Contracts*.

- **PPI Configuration:**

- A **Configured PPI** specifies that a PPI has been selected to be deployed for a given process variant, regardless of its optional or mandatory status. This attribute is a Boolean, whose value is true if the PPI is configured, (cf. Figure 27 for concrete syntax and cf. Figure 26 for abstract syntax).

The other 3 attributes are *string* and describe a PPI giving it a **name**, a **long name**, and a **description** about the target of the PPI.

4.4.3 Direct Relation Between PPIs

The selection of a PPI may imply the inclusion or exclusion of another PPI according to stakeholders' requirements.

- **Include** refers to the configuration of a PPI when another PPI is configured. In other words, it corresponds to the selection of a PPI (B) member of a *family of PPIs* during the configuration phase due to the selection of a PPI (A), i.e., if a PPI (A) is selected during the configuration phase, the PPI (B) has to be selected. For instance, if a distributor wants his customers to sign the subscription contract (*Number of Signed Contract* – PPI (A)), it is necessary to first send the designed contract to the customer (*Number of Sent Contracts* - PPI (B)).
- **Exclude** refers to the non-configuration of a PPI when another PPI is configured. In other words, the non-selection of a PPI (B) member of a *family of PPIs* during the configuration phase due to the selection of a PPI (A), i.e., if a PPI (A) is selected during the configuration phase, the PPI (B) should not be selected. For instance, the PPI(A) *Number of*

Signed Contracts by Portal excludes the PPI(B) *Number of Signed Contracts by Email*, if a distributor wants his customers to sign the subscription contract only by a portal. In this case, it is necessary to rule out the possibility to sign the subscription contract by email.

4.5 PPICT METAMODEL INSTANCE

The PPICT metamodel formalizes the PPICT considering the *measure type*, *measure representation*, *measure aggregation* and *PPI type*. In Figure 28, we show an instance of the PPICT metamodel based on the following definitions and rules that allow the modeling and management of PPI Variability:

- I) The definition of the PPICT.
- II) The definition of PPI reference.
- III) PPI variant definition.
- IV) Definition of the relation between PPIs.
- V) Rules of belonging of PPI variant and PPI reference.

Figure 28 shows a PPI family called *Number of Contracts*, where there are 4 levels of hierarchy. The root PPI *Number of Contracts* has 4 PPI variants related by a **Complex constraint**. Likewise, the PPI *Number of Sent Contracts* has 2 variants related by an **Overlap constraint**. The PPI *Number of Signed Contracts* has 2 variants related by a **Disjoint constraint**. Each of the individual relationships in a continuous line refers to a **Depend on constraint**. In addition, the dotted lines refer to a **Based-on constraint**. The PPI *Number of Active Signed Contracts* depends on the PPI *Number of Active Contracts* and is based on the PPI *Number of Signed Contracts*.

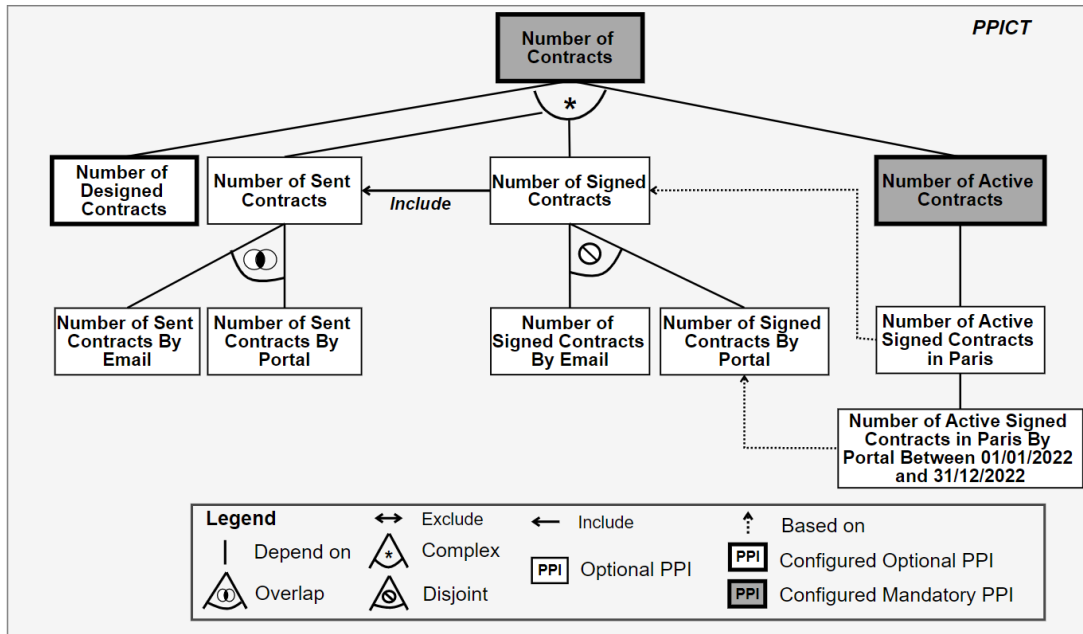


Figure 28 PPICT Metamodel instance

All PPIs in the *Number of Contracts* family have a *Number* value as well as measurement type and an aggregation type *COUNT()*. The PPIs *Number of Contracts* and *Number of Active Contracts* are the only two mandatory PPIs which have been configured. Most of the other PPIs are optional and have not been configured. The only optional PPI that has been configured is the *Number of Design Contracts*. The only **Include** relation in our example is that of the PPI *Number of Signed Contracts*, which, if configured, implies the configuration of the PPI *Number of Sent Contracts* in the same way. The **Time condition** is represented in the PPI *Number of Active Signed Contracts in Paris by Portal Between 01/01/2022 and 12/31/2022*. The **State condition** is represented in the PPI *Number of Active Signed Contract in Paris*.

In this chapter we described the PPICT Metamodel and the PPICT. We also defined the basis for our next contribution detailed in the next chapter, which presents the Process Performance Indicator Calculation Method (PPIC Method).

4.6 PPICT CONCLUSION

The PPICT facilitates PPI queries, because the new variant PPIs will have the correct query structure. It allows us to use the right attributes, tables, joins and conditions. Having a common structure between PPIs optimizes the calculation-time of new PPIs. The PPICT gives a partial view of the data model allowing users to

understand how tables are joined and which attributes are used for aggregations and joins. The PPICT also permits organizations to capitalize the modeled PPIs and provides correct syntax for new developers. PPICT is a language designed for decision-makers and PPI developers in order to analyze a process variant and to model and calculate PPIs.

Chapter 5: Process Performance Indicator Calculation Method (PPIC Method)

This chapter presents the Process Performance Indicator Calculation method (PPIC Method) and the PPIC metamodel which is an extension of the PPICT metamodel presented in chapter 4. The PPIC method extends the aforementioned BPFM method (Cognini et al., 2016) by integrating design stages to model and calculate PPIs within Customizable Process Models.

5.1 PPIC METHOD STAGES

The PPIC method describes the process that must be followed to design, model and calculate the PPIs linked to a Customizable Process Model. In this section, we are going to detail how the PPIC method supports PPI Variability modeling through five design stages:

- I) PPICT design.
- II) PPI design.
- III) BPFM and PPICT association.
- IV) PPICT configuration.
- V) PPICT-BPFM configuration checking.

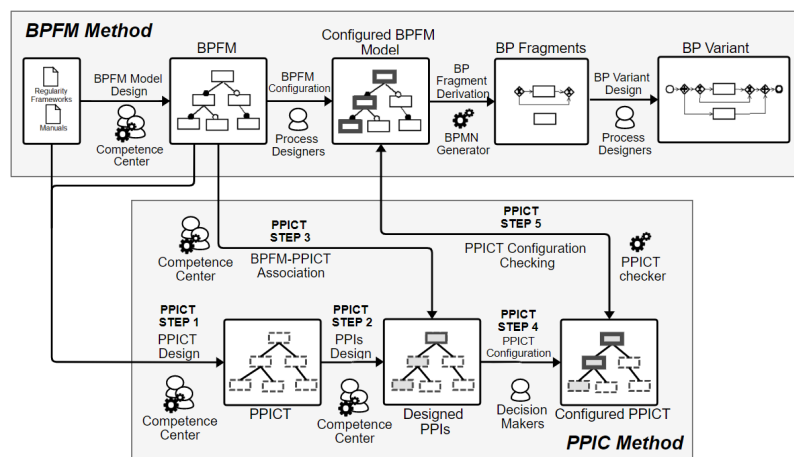


Figure 29 PPIC Method steps (bottom box) and BPFM Method (top box)

In each step, we will systematically use the running example Create Contract Process. This process is described with nine activities, two of which are sub processes. We model this process using BPFM notation (cf. Figure 30), which could be deployed differently according to the configured activities. To create a contract, it is necessary to design the contract, i.e., collect user information and fulfil their conditions. Then the user could sign or not the contract sent by mail or by the portal to finally activate it.

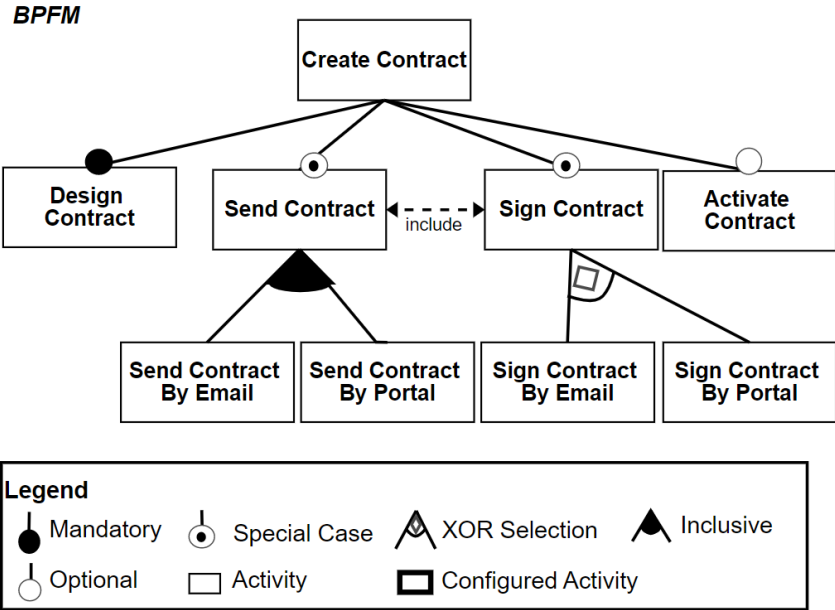


Figure 30 Create Contract Process modeled using BPFM notation.

Step 1, PPICT Design: refers to the manual addition of all PPI family members using the PPICT metamodel. PPICT Design allows the representation PPI 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 PPI family.

In this step, it is essential to know which activities could be used in the process that we want to evaluate. The BP designed adds each PPI to the PPICT using the *depend on* and *based on* constraints seen in the chapter 4. Given that the PPICT has been designed following our *PPI Variant* definition, it is necessary to have the tables used by each process activity as an input. To do this, process designers must know the data model and the link with activities or have access to the *process execution record* detailed in step 3. Below are the stages to design a PPICT.

- I) Place the PPIs according to PPICT definition seen in chapter 3 and use as a base the Query Tree to model the *family of PPIs*.
- II) Assign the right constraint to relate reference and variant PPIs.
- III) Choose the *Measure Type* of the PPIs to determine how PPIs are calculated depending on the result that the client is looking for, e.g., Number, Percentage, Proportion, Delay and Respect Rate Type.
- IV) Choose the *Measure Representation* to visualize PPI's tuples, e.g., value, listing, geographical and chart representation.
- V) Choose the *Measure Aggregation* to aggregate PPI's tuples, e.g., count(), max() etc.

In this step, we design the PPICT in the following ways:

- I) By adding the PPI root into the PPI family tree to evaluate a BP family.
- II) By adding new PPI variants of existing PPIs according to stakeholders' requirements.

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*, i.e., the value of the PPI.

A tree like the one illustrated below in Figure 31 is the output of this stage and the first step to build a PPICT.

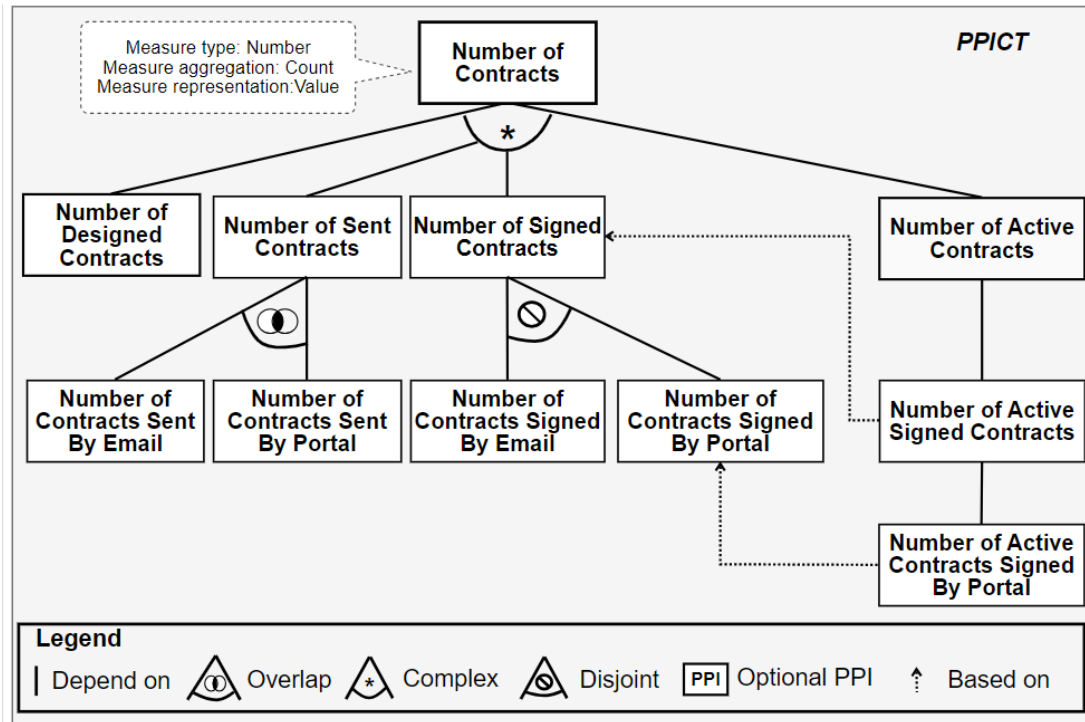


Figure 31 Example of step 1: Family *Number of Contract*

Step 2, PPIs Design: in this step all PPI family members must be designed according to stakeholders' definitions and design criteria as detailed in the *PPI class* in chapter 4. In this stage the competence center, which includes domain experts, BP designers and PPI calculation experts must make the following decisions design:

- I) *Inclusion* of PPI (B) due to the configuration of PPI (A)
- II) *Exclusion* of PPI (B) due to the configuration of PPI (A)
- III) *Optional PPI* to specify if a PPI could be configured optionally (white box in Figure 32)
- IV) *Mandatory PPI* to specify if a PPI must be configured by default (gray box in Figure 32)

Example: we have two mandatory PPIs, and the rest are optional PPIs, cf. Figure 32. In our example, we have only inclusion, if the PPI *Number of Sent Contracts* is deployed, the PPI *Number of Signed Contracts* must be deployed too.

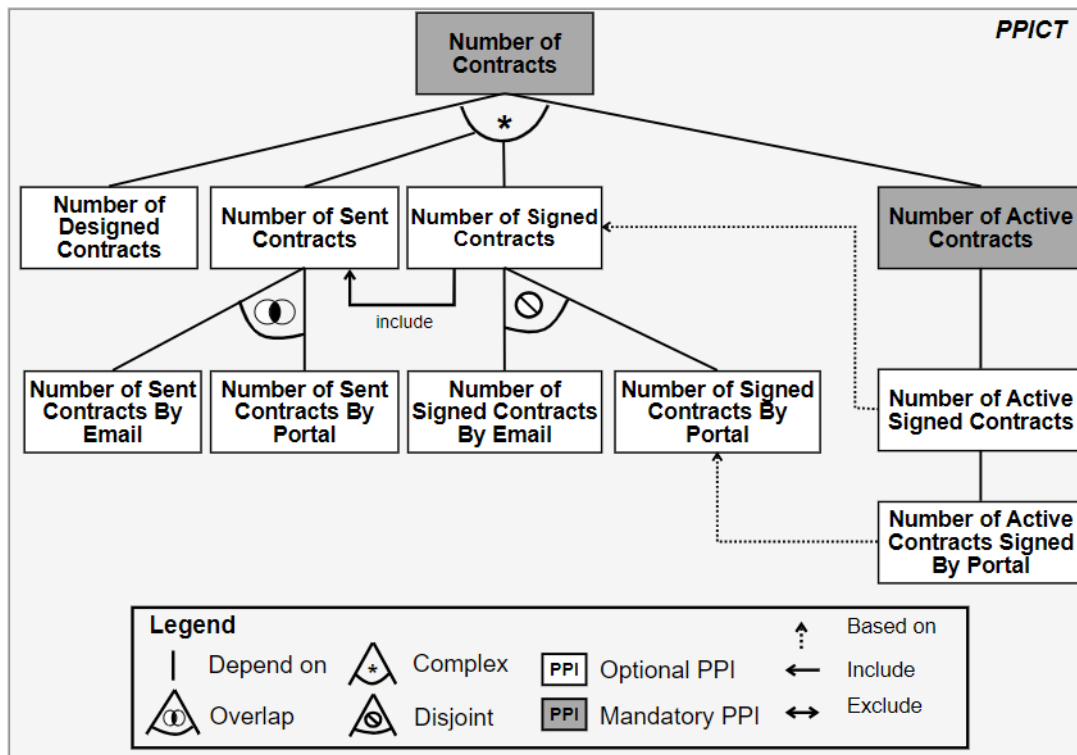


Figure 32 Result of step 2 PPI Design – Mandatory PPI (natural language view)

Step 3, BPFM-PPICT Association: enables the BPFM and PPICT association. This stage uses the PPI-Activity Mapping constraint of the PPICT relying on the BPFM Model to associate all PPIs of the tree, i.e., all PPI Family members, to the activities of the process model. This stage must be done manually using the PPI-Activity Mapping constraint.

Example: Figure 33 (Left part) shows all the activities of the *Create Contract process family*, which are described using BPFM notation (Cognini et al., 2016). Figure 33 (Right part) shows all PPIs designed to evaluate the *Create Contract* process family. The Mapping PPI-Activity (M) is shown here, e.g., between the PPI *Number of Contracts* and the activity *Create Contract* or between the PPI *Number of Active Contracts* and the activity *Activate Contract*. The mapping PPI-Activity is 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. This information can be found through the *Process Flow Execution Record* present by default in some software applications such as the INCOM'S one, cf. Figure 34. This record contains all tables and attributes used by each activity in a process. Sometimes this record could include the attributes used to execute the process. This record is detailed in section 5.2, Table 13.

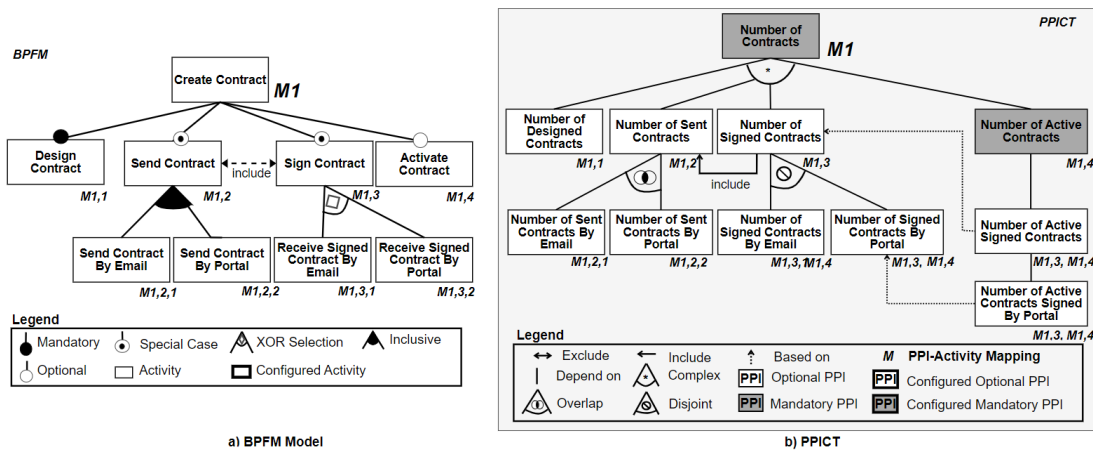


Figure 33 Example of BPFM-PPICT Mapping (M)

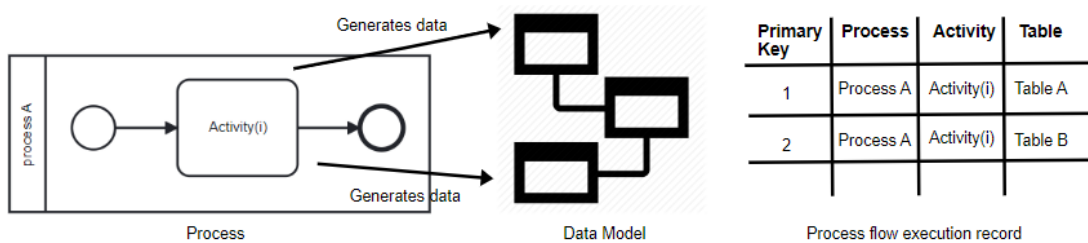


Figure 34 Example of Activity generating data

When the mapping between PPIs and activities is done, all PPIs must be calculated using languages that allow it. This is possible because the link PPI-Activity was made to identify the tables and attributes to use. For instance, a PP ICT like the one illustrated in Figure 35 uses SQL queries to calculate the PPIs.

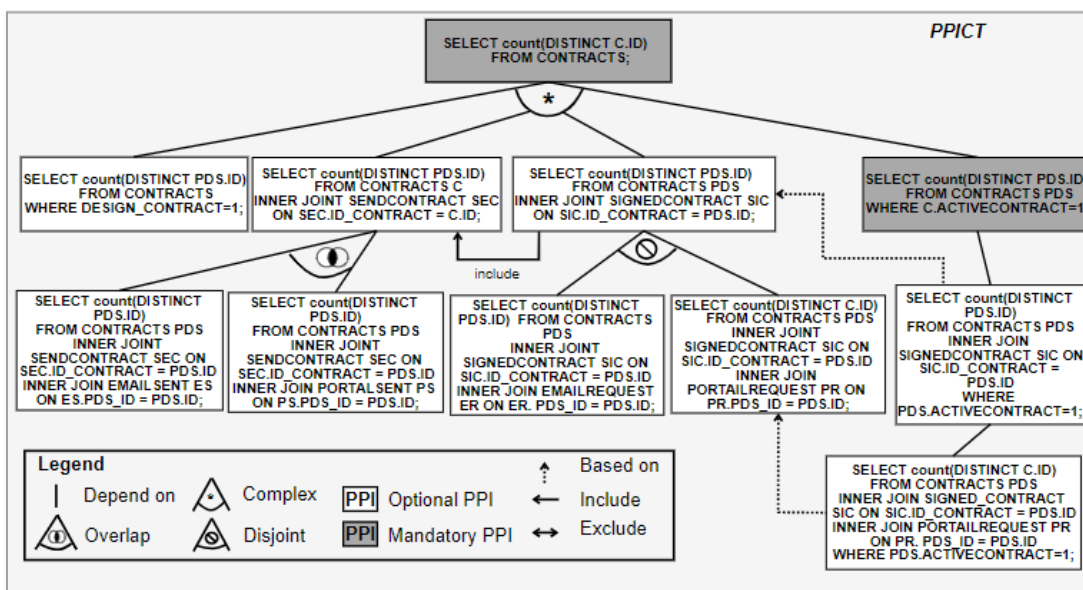


Figure 35 Example of PPI design (SQL view)

Step 4, PPICT Configuration: defines the configuration of PPI family members by the client. The configuration of a PPI depends on:

- I) Mandatory indicators required by regulatory entities. PPI PPI
- II) Stakeholders' requirements to evaluate their BP variants.

The PPICT Configuration allows the definition of PPIs family members that need to be included into the BP variant considering business regulations and decision makers' criteria. This step is done semi-automatically since mandatory PPIs are automatically configured. Optional PPIs must be manually configured, i.e., manually chosen. The configured PPI at this stage could be deployed if there is a PPI-Activity match according to the BP variant that has been configured.

Example: at this stage, decision makers configure the PPIs that they consider convenient to evaluate their process variants according to stakeholders' definitions and criteria. Figure 36 shows three configured PPIs (**rectangles with a thicker border**): *Number of Contracts*, *Number of Active Contracts* and *Number of Designed Contracts*. Decision makers can choose any optional PPI.

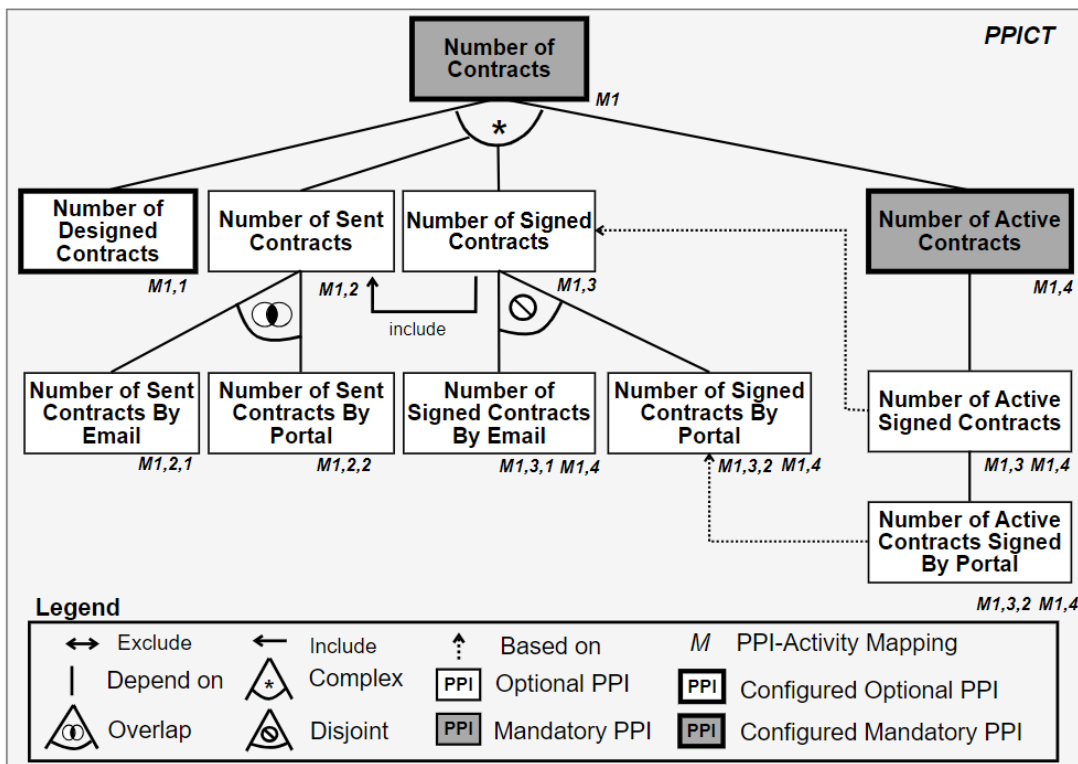


Figure 36 PPICT Configuration: configuration of *Number of contracts*, *Number of Active Contracts* and *Number of Designed Contracts*

Step 5, PPICT Configuration Checking: aligns the PPICT configuration with the BPFM configuration, i.e., checks if PPICT configuration matches with BPFM configuration. PPICT Configuration Checking allows users to check which members of the configured PPI 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. Currently, the method does not prevent errors during the configuration phase, therefore if there is any misalignment between the PPICT and the BPFM configurations, it is necessary to return to the previous steps to check the alignment between PPICT-BPFM configurations.

Example: at this stage, the matching of PPICT with BPFM model configurations is checked. In Figure 37, the PPIs *Number of Contracts*, *Number of Active Contracts* and *Number of Designed Contracts* are configured. However, only the PPIs *Number of Contracts* and *Number of Designed Contracts* are mapped to the corresponding **configured** activities, i.e., those PPIs are mapped to the *Create Contract* activity and to the *Design Contract* activity. As for the PPI *Number of Active Contracts*, it is mapped to the **non-configured** *Activate Contract* activity. In this case, the competence center must reconfigure the BPFM to align configured members of each family. This task should be done interactively to configure all activities corresponding to the configured PPI. After the reconfiguration, Figure 38 (a) and (b) show a correct alignment BPFM-PPICT.

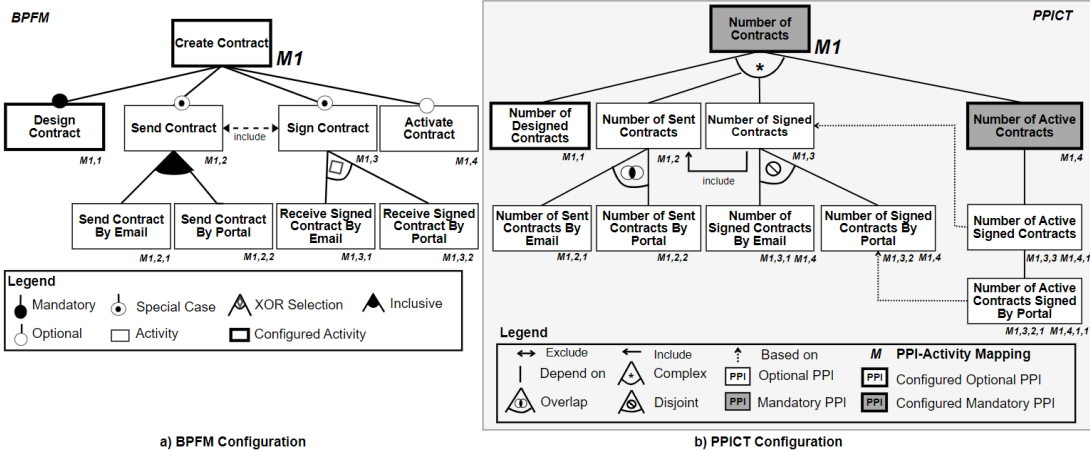


Figure 37 Miss-mapping between BPFM and PPICT because Activate Signed Contract Activity is not configured in the BPFM, but the *Number of Active Contracts* is configured

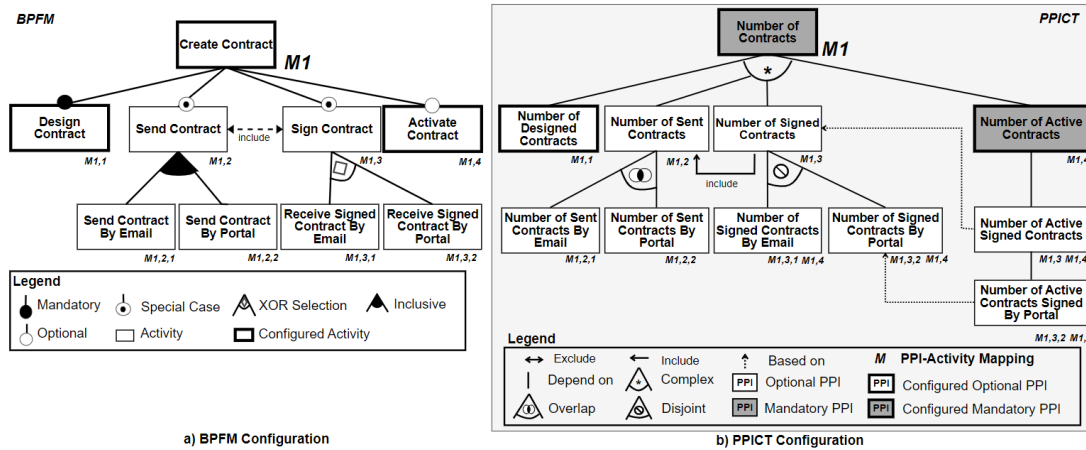


Figure 38 Correct BPFM-PPICT mapping: the PPI *Number of Active Contracts* is aligned with the activity *Active Contract*

Currently, during the PPI configuration phase, the activities are not configured automatically according to the configured PPIs. The configuration of the activities must be done manually. 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 by studying their alignment with the process activities in the BP family.

5.2 THE PPIC METAMODEL (LINK BETWEEN THE BPFM METAMODEL AND THE PPICT METAMODEL)

This section discusses how the link between the BPFM Metamodel and the PPICT Metamodel is implemented. We propose the PPIC Metamodel, which includes a class to identify the tables used by each activity of the process as well as a relation to link the PPICT metamodel and the BPFM Metamodel. These two classes are defined below.

Mapping PPI-Activity relation: this relation aims to link the PPI family modeling and the BP family modeling. 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. This construct is included in the meta-model of Figure 39. Its concrete syntax is indicated as an M in Figure 38.

Constraints class: this class is defined as an abstract class to centralize the multiple and binary constraints of the two metamodels.

Table Class: all PPIs must be calculated under a specific context, according to the tables used by each activity. For this, we propose to add a table class to the BPFM metamodel (refers to Figure 39 with a class corresponding to the list of tables used by each activity). The Activity-Tables relation is usually registered in the process flow execution record, presented by default in some software applications (cf. Table 13). Every single activity may generate essential data, which could be used to calculate a PPI. The PPICT metamodel assumes that the Activity-Tables relation is known and can be used to evaluate the process performance. A primary Key is used to identify independently the relation Process-Activity-Table.

Table 13 Process flow execution record

Primary Key	Process	Activity	Tables
1	Create Contract	Design Contract	Contracts
2	Create Contract	Send Contract	SendContracts
3	Create Contract	Send Contract by Email	SendContracts
4	Create Contract	Send Contract by Email	EmailSent
5	Create Contract	Send Contract by Portal	SendContracts
6	Create Contract	Send Contract by Portal	PortalSent
7	Create Contract	Sign Contract	SignContracts
8	Create Contract	Sign Contract by Email	SignContracts
9	Create Contract	Sign Contract by Email	EmailRequest
10	Create Contract	Sign Contract by Portal	SignContracts
11	Create Contract	Sign Contract by Portal	PortalRequest
12	Create Contract	Activate Contract	Contracts

Regarding the classes and relation used at each phase of the method. In step 1, the competence center has to use the constraint classes in the form of *Binary PPI* and *Multiple PPI* constraints as well as the PPI class. For step 2, the condition classes such as time conditions and entity conditions are used as well as the *measure aggregation*, *measure representation* and *linear evaluation* classes. For Step 3, only the relation *Mapping PPI Activity* and the class tables are used. The attribute *PPI configuration* of the PPI class is used in step 4. Finally, step 5 uses the relation *Mapping PPI Activity* and the *Table* class.

PPIC Metamodel : Link between BPFM Metamodel and PPIC Metamodel

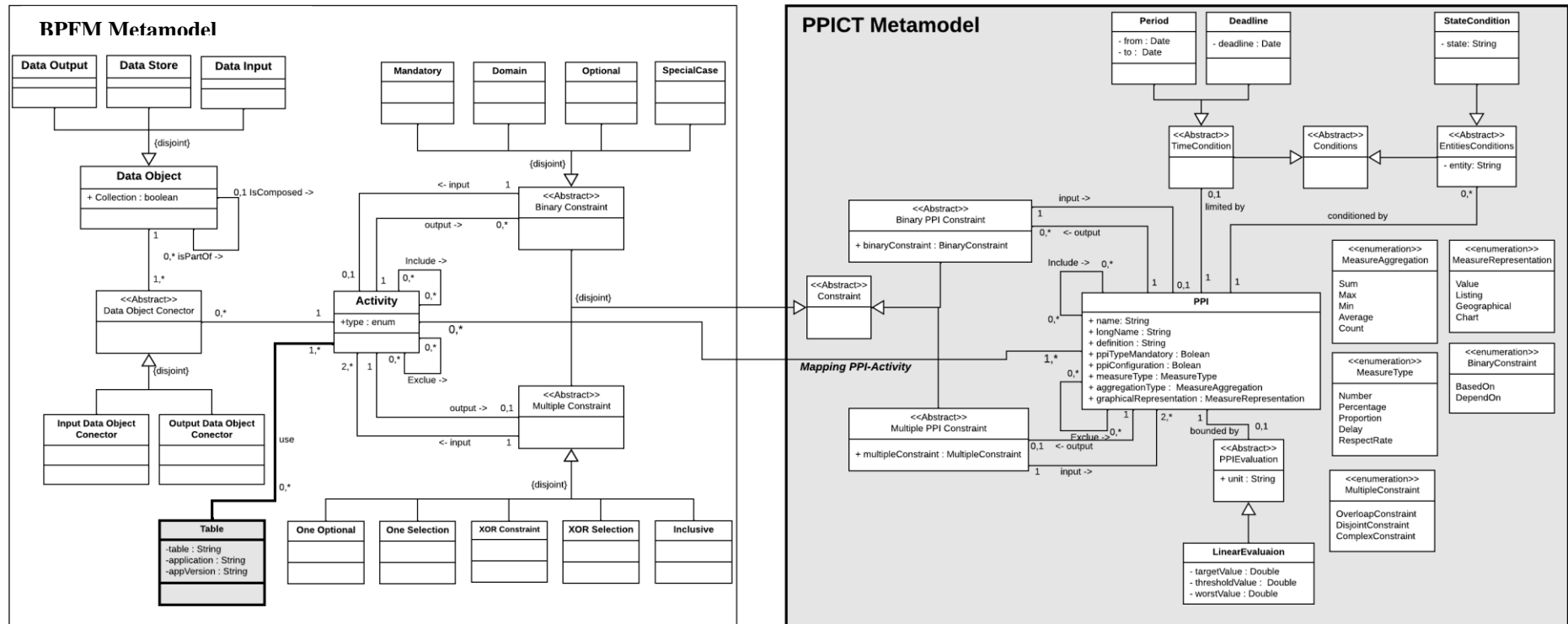


Figure 39 PPIC Metamodel: link with the BPFM metamodel and PPIC Metamodel.

5.3 EVALUATION OF THE PPIC METAMODEL

To evaluate our propositions of PPICT and PPIC Methods, we have used two groups of criteria identified in the state of art, namely for the group 1 the integration level between PPIs and Customizable Process Models and for the group 2 the expressive capacity of PPIs.

Group 1: The integration level between PPIs and Customizable Process Models.

I) Model PPIs independently of the language used to model the BP.

The PPICT supports this requirement as it allows the modeling of PPIs based on a language different than the language to model Business Process. This is to model the PPI Variability after making the link with the Customizable Process Model.

II) Model PPI Variability together with Customizable Process Models.

The PPICT and the PPIC Method support this requirement allowing the design of PPI Variability and Customizable Process in a single model to create a synergy between Customizable Processes and process performance indicators.

III) Model the PPI-Activities relation of a business process.

The PPICT and the PPIC Method support this requirement allowing the modeling of the link between PPIs and the activities to identify the resources that generate the data used to calculate indicators.

Group 2: The expressive capacity of PPIs.

IV) Define a variable performance indicator.

The PPIC Method supports this requirement allowing the definition of PPI Variability, establishing how it impacts the business process evaluations, and how to deal with variable PPIs.

V) Model the relation between PPIs.

The PPICT and the PPIC Method impact this requirement because they model the relationship between PPIs and define how PPIs are related to each other.

VI) Consider PPI Variability by restriction.

The PPICT and the PPIC Method support this requirement allowing the modeling of PPI Variability when their variants are known a priori. That is, when the PPI template has all the known customization possibilities.

VII) Consider PPI Variability by extension.

The PPICT and the PPIC Method support this requirement allowing the modeling of PPI Variability when their variants are not known a priori, that is, when the PPI template does not have all the PPI customization possibilities.

In conclusion, this chapter presented the PPIC Metamodel specifies the link between the PPICT metamodel (on the right side in the Figure 38) and the Customizable Process Model Metamodel, i.e., BPFM Metamodel (on the left side in Figure 38). This link is composed of the Constraints class, the Mapping PPI-Activity class and the Table class. The next chapter describes the user validation of our method.

Chapter 6: Validation of the Process Performance Indicator Calculation Method Through a User-Centered Evaluation

This chapter presents the evaluation of the Process Performance Calculation Method through a user-centered evaluation using the THEDRE method (Mandran & Dupuy-Chessa, 2018). For this validation we have set an experimental protocol of 10 interviews. The goal consists of measuring and validating the PPI development among INCOM's engineers and novice students when using the PPIC Method. The experimental protocol is detailed in Appendix A.

6.1 EXPERIMENTAL PROTOCOL DESIGN

The *Design Experiment's* step in THEDRE refers to the definition of the experimental protocol and the type of results that we want to analyze (cf. Figure 40). Steps *Build Experiment* and *Build Actionable Tool* refer to the development of the experimental guide (interviews and questionnaires) and the development of the experimental material (workbook). The experimental guide and material help to measure and control the implemented tests using an existing PPICT. Both correspond to the so-called actionable tools, cf. Figure 40. The step *Collect and Analyze Data* refers to the collection and analysis of the methods used, the knowledge and the know-how of the participants. Our qualitative experimental protocol uses an operational guide during the experiment to organize observations and collect data, taking notes of users' behaviors through brief discussions.

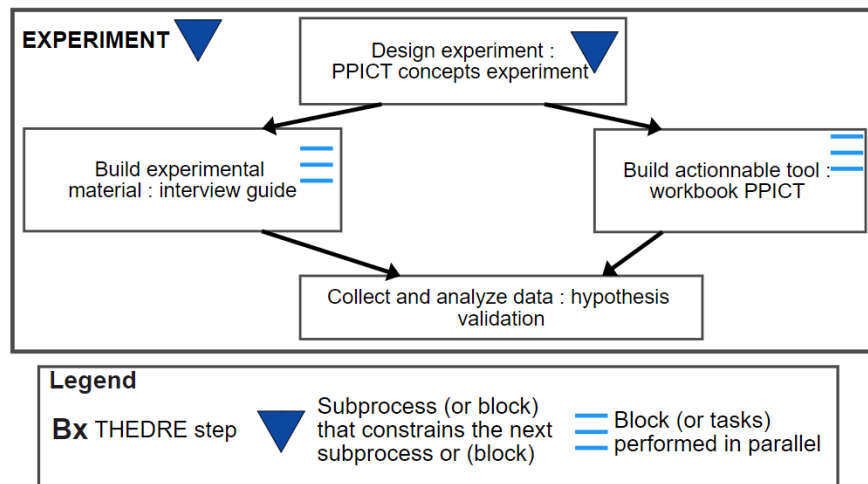


Figure 40 Experimental steps instantiating the Experiment phase of the THEDRE method.

We have set the following experimental targets:

- I) Involve indicator developers in the modelling of PPIs linked to Customizable Process Models.
- II) Explore how users express their PPI definitions and calculate the variable PPIs in the Customizable Process Models.
- III) Identify users' modeling methodologies and practices of calculation and recalculation of PPIs according to the deployed Customizable Process Model.

The knowledge and skills of the participants depend on their experience. That is why we carried out 10 individual experiments to avoid any influence between participants: 5 with novice participants and 5 with expert participants. Each experiment was carried out in 6 states and 3 phases where the goal of each stage was to answer a questionnaire to validate the 8-hypotheses detailed below. The execution of the experiment protocol includes an explanation phase, a practice phase, and a discussion phase. All the experimental protocol was designed to bring the information necessary for the validation of the 8 hypotheses. For each experiment, the qualification of the questionnaires was evaluated to improve and develop a software tool which will be explained in the next chapters.

We have fixed the following hypotheses (H) to evaluate during the experimental protocol execution:

- H1: Participants do not have a formalized method to calculate the PPIs.

- H2: PPIs are impacted by Process Variability and cause uncertainty on the PPI calculations.
- H3: The PPIC Method helps participants to differentiate the main concepts of the PPCIT.
- H4: Participants can use the PPICT as well as understand the relationships and constraints between PPIs.
- H5: PPICT allows participants to place new PPIs in the tree structure.
- H6: The PPIC Method helps participants to fill all PPI attributes of the PPICT.
- H7: Existing PPIs allow participants to create new ones easily.
- H8: The PPICT helps to understand the data structure.

Participants should evaluate the concepts of the PPIC Method and the definitions of the PPICT proposed to validate the 8-hypotheses presented before. During the discussion phase of our qualitative experimental protocol, experts and novices expressed their individual points of view. They were also encouraged to propose improvements and raise questions.

Our experimental protocol has been structured according to the MATUI tree, which is part of the THEDRE method (Mandran & Dupuy-Chessa, 2018) to lead research in human-centered computer science and guide researchers (cf. Figure 41). MATUI helps to guarantee the traceability of experimental works. Regarding our context, we have chosen the pink way of the MATUI tree (indicated with thick arrows in the figure) because we have a dynamic experimental material, i.e., an experimental material divided into two parts: interviews and tests in a real case without simulation (called according to the authors an *Activable Component AC*) (cf. Figure 41). The experiment description of the chosen tools and produced data was essential to develop a relevant experimental protocol as well as share a common vision of PPI Variability among researchers, experts, and novices. The result of this development was an experimental protocol divided into 3 workbooks. cf. Appendix A, Appendix B, Appendix C.

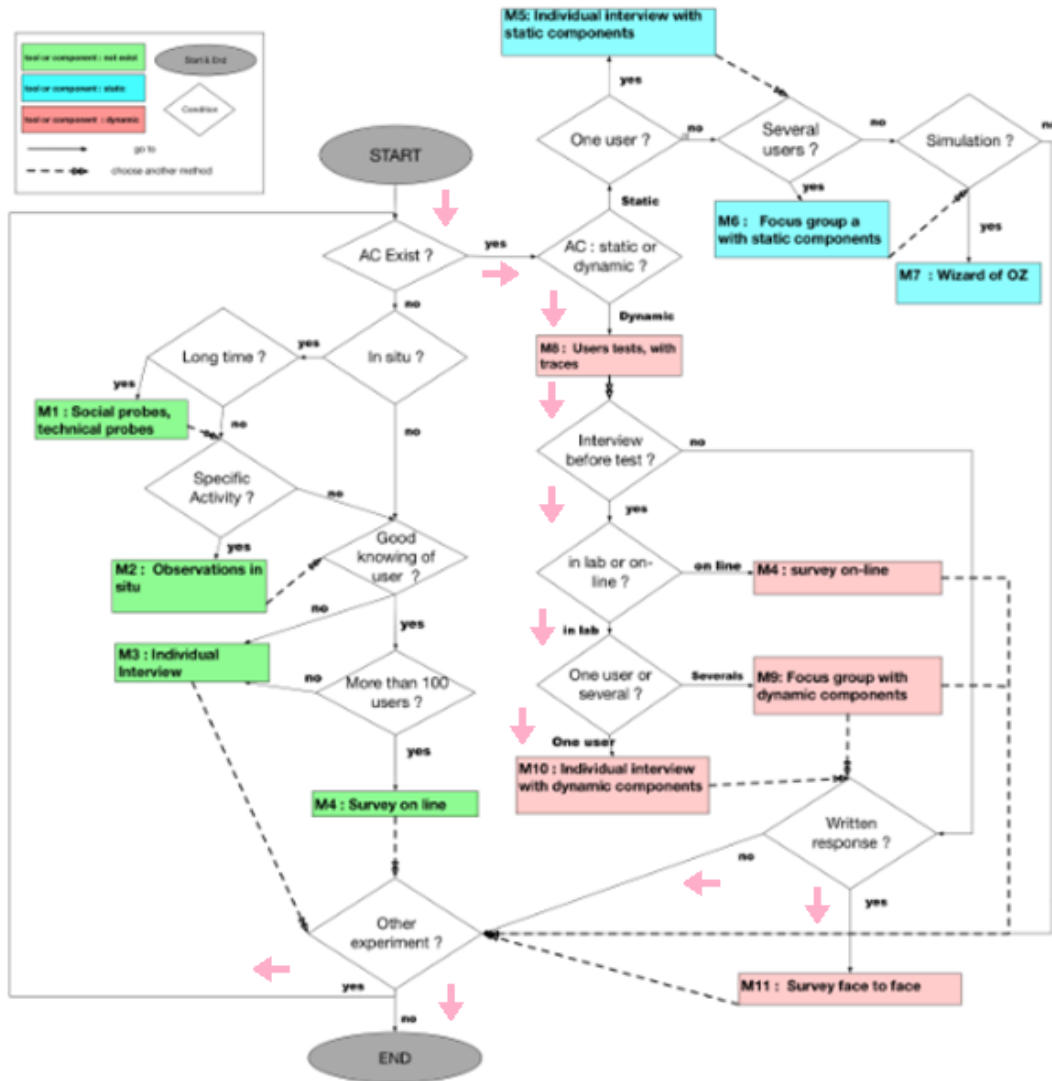


Figure 41 MATUI decision tree (Mandran & Dupuy-Chessa, 2018), pink way in our case due to a dynamic experiment

6.2 PARTICIPANTS

As mentioned before, users were divided into two groups: experts and novices. 10 individual evaluations were carried out concerning 5 INCOM experts and 5 novice students, French spiker only. We consider as experts, participants who have more than 5 years of experience in the development and exploitation of databases, development and implementation of business processes as well as some experience in business intelligence. We consider as novices, participants who have fundamental knowledge of business processes and databases, such as second-year undergraduate students in information systems.

Table 14 Professional experience of participants

Participant	Expert Participant	Novice Participant
1	31 years of Professional experience	-
2	37 years of Professional experience	-
3	5 years of Professional experience	-
4	6 years of Professional experience	-
5	5 years of Professional experience	-
6	-	2 years of Professional experience
7	-	3 months of Professional experience
8	-	1 years of Professional experience
9	-	6 months of Professional experience
10	-	4 months of Professional experience

6.3 INSTRUMENTS USED DURING EACH EXPERIMENT

From the beginning of the execution of the experimental protocol, users' perspectives must be collected. Each experiment is based on tests, measures, surveys, observations, interviews, questionnaires and artefacts for data collection and discussion about participants' PPIs development techniques. For the execution of the experiment, it is necessary to have an existing PPICT. We used questionnaires 1, 2, 5 and 6 shown in Appendix A to obtain the results of experiments. For example, Figure 42 shows a part of questionnaire 2. Questionnaires contain closed and open questions. At the end of the exercise, a discussion enabled us to gather critical comments from participants about the effectiveness, usefulness and relevance of the explanation and the material used for the experiment. During the interview, we were open to any remarks or observations expressed by the participants in the experiment. In addition, to analyze the achievement of the goals of section 1, we analyzed the results of the questionnaires in two groups (expert and novice answers).

Q2: Context of the experiment

Interview: Diego takes notes and records

I have just introduced you to the notion of variability of indicators,

- What do you think of this notion of variability of indicators?

- Have you ever encountered this problem of indicator variability?

- What solutions did you use to address this problem?

Figure 42 Example of experimental material (Questionnaire)

Regarding our qualitative goals to determine whether the PPICT concepts are clear (hypothesis 4), we implemented a workbook, (cf. Figure 43), along with an oral explanation of PPI concepts and PPI family model, (cf. Appendix B). This workbook aims to build a PPICT on a new case of study *Number of Meters* using the PPICT method. This workbook was designed in French because the experiment took place with French-speaking participants. During the experiment, participants could interact with the PPICT. Before the test, the animator of the experiment explained the methodology and the concepts to be evaluated. Thus, participants could contribute by proposing modifications and/or improvements to the PPICT and the PPICT Method.

Regarding the first exercise, (cf. Figure 43), the participant must execute each step numbered from 1 to 6, which will allow him or her to complete the proposed PPICT using the legend and the rules of the PPICT addressed during the animator's

presentation (cf. Appendix B). The participant will have the necessary information to build, modify and complete the proposed PPICT.

PPICT

1. Place the three indicators in the PPICT.
2. Modify the Symbols of the model if necessary.
3. Specify the indicator Number of meters by manufacturer as mandatory.
4. Specify the other two flags as optional.
5. Configure mandatory indicators for PPICT.
6. Configure the optional PPICT indicators associated with the diameter.

Nombre de Compteurs par Fabricant

Nombre de Compteurs posés actuellement dont la date de pose est inférieur ou égal au 31/12/2004

Nombre de Compteurs par Diamètre posés actuellement

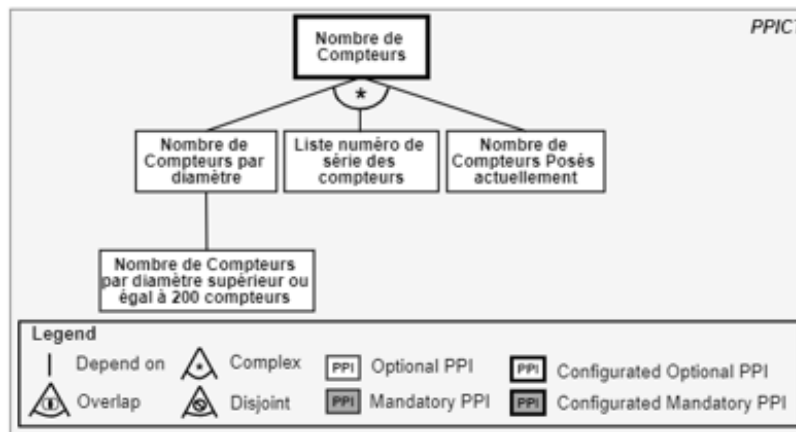


Figure 43 Experimental Material: workbook

We have also designed different matrixes to evaluate the methodological aspects and proposed concepts, e.g., Figure 44 (cf. Appendix A questionnaire 3 and 4). Those matrixes help us get feedback on proposed concepts, which were evaluated in terms of proportion and ratios. The ratios also allow us to compare different experiments by classifying the concepts by hypothesis and by clarity.

Clarity: 1 not at all clear... 4 quite clear.

Concept	Definition	Clarity
PPI	Optional Indicator Not Configured	1 2 3 4
	<i>Comments:</i>	

Figure 44 Matrixes to evaluate the methodological aspects and proposed concepts.

Each experimental guide and experimental material help to validate the hypothesis presented in section 6.1. The questionnaires and the workbook allow us to know if the participants understood the main concepts of the PPICT and the rules of the PPIC method. Below, Table 15 details which instruments were used to validate or not the proposed hypotheses.

Table 15 Experimental material and experimental guide that help to validate hypotheses.

Hypothesis	Instruments: Experimental material and Experimental guide that help to	Goals and exercises
H1: Participants do not have a formalized method to calculate the PPIs.	Questionnaire 1 of the experimental guide.	Identify the practices and methodologies used by the participants to calculate the indicators.
H2: PPIs are impacted by Process Variability and cause uncertainty on the PPI calculations.	Questionnaire 2 of the experimental guide.	Know if the participant understood the context of Customizable Process Models and PPI Variability.
H3: The PPIC Method helps participants to differentiate the main concepts of the PPCIT.	Questionnaire 3 of the experimental guide.	Know if the participant understood the PPICT definitions.
H4: Participants are able to use the PPICT as well as understand the relationships and constraints between PPIs.	Questionnaire 4 and 5 of the experimental guide and workbook of the experimental material.	Use a workbook to roll out the PPIC method and to interact with the PPICT by applying the definitions.

H5: PPICT allows participants to place new PPIs in the tree structure.		Identify if the participant manages to carry out the exercises of placing the new PPIs inside the PPICT and writing the queries of PPIs added.
H6: The PPIC Method helps participants to fill all PPI attributes of the PPICT.		
H7: Existing PPIs allow participants to create new ones.		
H8: Understanding the data structure owing to the PPICT.	Questionnaire 5 and 6 of the experimental guide and the workbook of the experimental material	

6.4 EXPERIMENT ANALYSIS

The research targets were checked regarding the methodological hypotheses presented in section 6.1. This step was crucial to reason about the development of our software tool because it allows us to identify improvements and focus on users' needs. Regarding the experimental protocol results, experts and novices were able to use the PPIC method according to the PPI Variability modeling language proposed. During the experiments, each hypothesis proposed some exercises based on a PPI family scenario supported by interview guides and workbooks. Thus, experts and novices expressed their points of view individually on either improvements or questions. A synthesis of the eight hypotheses is presented below:

- H1: Participants do not have a formalized method to calculate the PPIs.

Observation: none of the novices and experts have formal methods for calculating PPIs, but all experts do have practices used independently based on their experiences. Four experts apply software development methods for PPI calculation. 9 out of 10 participants said that the PPIC method covers this lack by giving a route plan to model and calculate PPI Variability linked to Customizable Process Models. Experts mentioned that *“the complexity of controlling process changes without proper modeling, induces recalculating the PPIs”*. Moreover, experts and novices agreed that *“currently there is a lack of reliability in the calculation of PPIs because they do not*

have tools or methods that allow them to prune and calculate PPIs". We therefore consider this hypothesis as validated.

- H2: The PPIC Method helps participants to differentiate the main concepts of the PPCIT.

Observation: 9 out of 10 participants agree that thanks to the PPIC method they were able to identify the constraints and rules to build a PPICT. 4 experts think that the concepts of the PPICT are clear after having used the PPIC method especially to model relationships between PPIs. All experts agree that the impact of process Variability concerns PPIs. One of them said that *"the standardization of a software product is very complex. However, with the PPICT and the PPIC Method it could be possible"*. During the experimentation, novices were able to identify how customizable processes are modeled and how PPI Variability was approached using PPICT concepts. All participants were aware that software publishers need to deal with this problem with a model such as the PPICT. That is why this hypothesis is also considered as validated.

- H3: Participants differentiate the main concepts of the PPCIT.

Observation: 3 out of 5 Experts and 5 of the 5 novices said that *"it is possible to understand the main definitions of PPI, reference and variant PPI"*. However, 2 experts propose that *"It could be necessary to divide the definitions of an invariable PPI and a variable PPI"*. 2 experts and 2 novices preferred simpler definitions like *"Father PPI"* and *"Son PPI"*, but they consider that *"the main concepts of the PPICT are clear especially the tree and the rule to build it"*. That is why we consider this hypothesis as partially validated.

- H4: Participants can use the PPICT as well as understand the relationships and constraints between PPIs.

Observation: when new variants PPIs are added, 4 experts and 5 novices said that *"The PPICT is clear and easy to use"*, *"between brothers it is clear and consistent"*, 4 experts and 4 novices said that *"the PPICT had allowed them to save time in the calculation of PPIs"*, all participants said that *"the PPICT has allowed them to know and manage the PPIs information to calculate them"*. Nevertheless 3 experts and 4 novices 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"*.

2 novices said that *“it had been difficult to configure a PPI because of the black box, it would be nice to think of a color”*. This improvement is relevant for PPICT to configure a PPI, which is why we have implemented this functionality on the PPICT tool. This hypothesis is therefore validated.

- H5: PPICT allows participants to place new PPIs in the tree structure.

Observation: all novices and 4 out of 5 experts appreciated the notation and considered it understandable. However, one expert said that *“it was difficult to place new PPIs when the labels were too long,”* 3 experts and 4 novices said that *“it was difficult to add a new flag when you have two possible parents”*. To overcome this problem, we have included in our PPICT approach the relation "based on" which allows users to place a PPI when there are two possible reference PPIs. One expert proposes to suggest the possible reference PPI of a new variant PPI using the labels through a search function. This functionality could be implemented in future research. This hypothesis is therefore considered as validated.

- H6: The PPIC Method helps participants to fill all PPI attributes of the PPICT.

Observation: 4 experts and all novices 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”*. All novices agree that the method allows them to identify the PPIs on which novices should be based to add new PPIs. Thus, it allows them to choose the right attributes according to the reference PPI. One of the experts mentioned the importance of having a method to guide users when identifying the steps to follow to calculate new PPI variants including right attributes. *“If the right PPI Reference is used, the attributes are already included. This means that no new ones need to be included, except for a specific join”*. This hypothesis is validated.

- H7: Existing PPIs allow participants to create new ones easily.

Observation: All participants concluded that *“the more information we have in the PPICT, the easier it is to add new PPIs.”* All experts think that the existing PPIs provide fundamental information like attributes, joins and tables to calculate new PPIs. According to experts, it is difficult to know the whole database. Having existing PPI allows experts to recontextualize their queries and choose the best joints calculate other

PPIs. The novices think that it is an excellent way to know exactly the tables and attributes that must be used when adding a variant, since the reference PPI is already described in a correct way using the right joints and syntax. This hypothesis is validated.

- H8: PPICT helps to understand the data structure.

Observation: 4 experts said that “*even if some PPIs were more complicated to calculate than others, the PPICT is an easy-to-use tool to build PPIs and it can be used by beginners or experts in PPI calculation*”. 3 Novices see this method as a solution to model and calculate indicators in general, even if they are not linked to a Customizable Process Model. 4 novices said that if they had known the PPICT and the PPIC method, they would have used them for modeling and calculating their indicators from the beginning of learning queries in relational databases. This hypothesis is validated.

Table 16 Synthesis of hypothesis validation

Hypothesis	Validation Status
H1: Participants do not have a formalized method to calculate the PPIs.	Validated
H2: The PPIC Method helps participants to differentiate the main concepts of the PPCIT.	Validated
H3: Participants differentiate the main concepts of the PPCIT.	Partially Validated
H4: Participants can use the PPICT as well as understand the relationships and constraints between PPIs.	Validated
H5: PPICT allows participants to place new PPIs in the tree structure.	Validated
H6: The PPIC Method helps participants to fill all PPI attributes of the PPICT.	Validated
H7: Existing PPIs allow participants to create new ones easily.	Validated
H8: PPICT helps to understand the data structure.	Validated

The experimental results and research targets were checked using methodological hypotheses such as clarity of concepts to determine possible improvements of the PPICT Method. All the hypotheses were totally validated except the hypothesis 3 that was partially validated. The prototype allowed us to identify the limits but also the advantage of the PPICT Method facilitating PPI Variability modeling linked to a Customizable Process Model.

6.5 LIMITATIONS IDENTIFIED DURING THE EXPERIMENT

6.5.1 Limitations of concepts and functionalities

We have attained some scientific knowledge from the experimental results, and we have identified some limits of our proposition. We have divided the limits in two dimensions, depending on the subject's expertise:

Table 17 Limits of the PPICT and PPIC Method

Participant Type	Limit	PPICT	PPIC Method
Experts	<i>"There is no view of the query on the same diagram"</i>	X	
	<i>"Classifying a PPI in a family too large is without doubt difficult"</i>	X	X
	<i>"Adding a new relation can be complicated if we don't know the data at all"</i>	X	X
	<i>"It is impossible to suggest automatically existing reference PPIs for new variant PPIs"</i>	X	
	<i>"The terminology is a little complicated; father and son can be an alternative for reference and variant PPI"</i>	X	X
	<i>"There is no way to model PPIs based on NoSQL databases"</i>	X	X
Novices	<i>"Being familiar with data models"</i>	X	X
	<i>"Making a mistake easily with the choice of the constraints"</i>	X	

To answer the limitations met when adding a new PPI when it has two possible reference PPIs, we have included the function “Based on” to the PPICT. The PPICT metamodel was designed to allow the definition of all the possible reference and variant PPIs added by restriction or extension. However, we do not consider PPIs related to process resources, because we have concentrated only on the tuples resulting from the execution of a business process and not from whoever executes them.

PPI Variability for NoSQL is still a question that we did not study in our research because currently we do not know if the inclusion of PPIs in the PPICT can be applied or not in NoSQL systems. The need to suggest possible reference PPIs identified in experiments is also left for further studies because this implies modifying the PPIC tool including an analysis of all possible references of a PPI regarding the nearest set of tuples inclusion rule. Based on the knowledge acquired during our research project, we think that a possible improvement of the PPICT tool could be the integration of data mining techniques to determine if a variant PPI is really derived from a given reference PPI. This integration could rely on the link between the used tables and attributes during the execution of the process and the PPI to be calculated.

The PPICT metamodel is supported by a software tool called ADOxx detailed in the next chapter. This tool can be used as a template-based model. Nevertheless, the usage of this formalism in real industries needs a software license, since ADOxx is only free for research goals. The main reason we used the ADOxx Metamodeling tool was that BPFM was already implemented on this open research platform, so we encountered fewer difficulties of interoperability to integrate, formalize and implement our PPIC method and PPICT metamodel. This allowed us to evaluate the behavior of our method implemented in a software tool.

6.5.2 Limitation of the experiment saturation

Glaser & Strauss (Glaser & Strauss, 1967) show that most of the interview saturations in qualitative experiments occurred by 12 individual experiments, i.e., 12 individual interviews. The study of (Lakens, 2022) shows that after interviewing 20 participants, no new categories emerge, reaching a saturation point. Additionally, the study of (Marshall et al., 2013) shows that a single case study should generally include between 15 and 30 interviews. The authors argue that 69% of all qualitative

information system studies are based on a single case and used fewer than 30 interviews. With 10 interviews made in our experimental protocol compared we score 6 more than the minimum of 4 interviews for a single case analyzed by (Marshall et al., 2013). (Marshall et al., 2013).

Table 18 shows the result of the comparison between the numbers of interviewees interviewed and the total time taken to interview the participants regarding a theoretical investigation, a theory research, single case research and multiple case research. This table also considers the studies most cited in google and other tools in general. Concerning our study, we have 6 interviews more than the usual minimum for a single case validation in information system literature. However, we have 2 interviews less than the minimum desired (Marshall et al., 2013).

Table 18 Interviewees, Interviews, and Contact Time by Research Design (Marshall et al., 2013)

	Type of research experiment in published articles			Top cited articles	
	Grounded Theory	Single Case	Multiple Case	Google Top Performers	Top Performers
Interviewees					
Minimum	6	4	10	7	12
Quartile 1	15	13	22	19	20
Quartile 2	27	24	39	29	24
Quartile 3	59	43	45	68	40
Maximum	200	200	74	105	105
Interviews					
Minimum	6	4	10	7	12
Quartile 1	17	13	22	19	20
Quartile 2	29	23	40	29	24
Quartile 3	59	45	49	68	40
Maximum	200	200	74	127	127
Contact Time (hrs)					
Minimum	5.5	4.0	6.0	5.5	12.0
Quartile 1	15.8	10.0	21.2	31.2	23.0
Quartile 2	37.3	28.0	38.8	39.0	37.3
Quartile 3	70.0	46.0	68.8	71.0	38.8
Maximum	222.3	222.3	100.5	222.3	222.3

Our advantage is concentrated in the number of hours per interview, which was an average of 3 hours. This means that 30 hours of experimentation were developed (10 interviews of 3 hours). This places us between the quartiles 2 and 3, cf. Figure 45.

This then allows us to be close to the value recommended by (Marshall et al., 2013) of 28 hours of interviews.

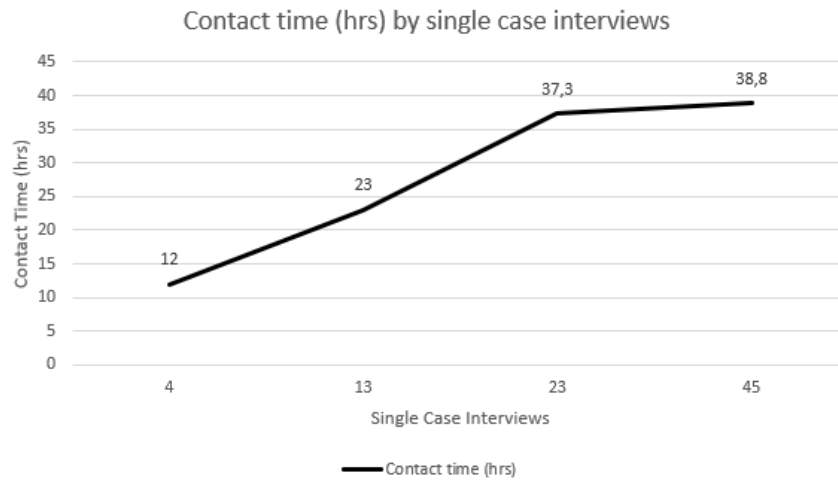


Figure 45 Contact time by single case interviews based on the analysis of (Marshall et al., 2013)

6.6 CONCLUSION OF THE USER-CENTERED EVALUATION

The execution results of our experimental protocol conclude on the success of providing a solid method and model for PPI Variability definitions integrated into Customizable Process Models. All participants agreed that our contribution reduces the PPI design-time and calculation-time contributing to the Customizable Process Model evaluation. Nevertheless, some terms such as reference PPI and variant PPI were too obscure for participants who suggested alternatives such as father PPI and son PPI.

We are aware that having completed 10 individual qualitative experiments was a limitation in our user-centered evaluation. Indeed, the COVID crisis has limited us in the development of other experiments in the same conditions as the others. (Glaser & Strauss, 1967) show that most of the data saturation for single cases in qualitative experiments had occurred at least 12 individual experiments, i.e., 12 individual interviews.

Chapter 7: Process Performance Indicator Calculation Tool (PPIC Tool)

This chapter presents our software tool developed on ADOxx⁹. The tool allows us to guide users in the development of PPIs using the PPIC method. The PPIC method is supported by a metamodel. We have implemented a graphical notation to illustrate and describe the PPICT. The PPICT has been developed by extending the BPFM tool, which was also developed on ADOxx. The development of our software tool included PPI developers, decision makers and business process managers who are the experts of the experimental protocol. To explain this tool, we illustrate its use through the 5 steps of the PPIC method presented in chapter 5.

Figure 46 shows the architecture of the ADOxx modeling tool. At the first level of the hierarchy, we found the Meta2 Model which was developed in C++ by the OMILAB¹⁰ group. This Meta2 Model is instantiated at level 2 of the hierarchy by the Meta Model of the modeling tool, here, the meta classes were instantiated during the development of our PPIC tool at level 3 of the hierarchy. At this level our PPIC metamodel was implemented separating the classes and relationships. All the rules that must be used in the modeling tool are implemented. In our case, the Development Tool was handled and used only by the Ph.D. student and the Modeling Tool was used by all the participants in the experiments and by INCOM's designers of processes and indicators.

⁹ <https://www.adoxx.org/live/home>

¹⁰ <https://www.omilab.org/>

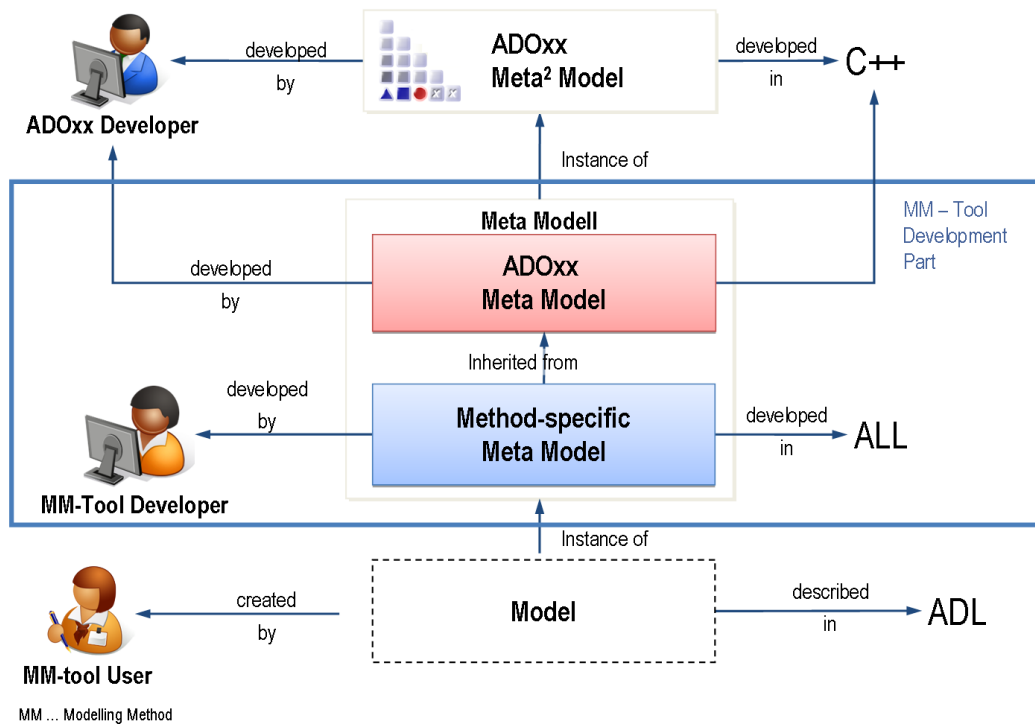


Figure 46 ADOxx Meta Modelling Platform Hierarchy ¹¹

7.1 USE OF THE TOOL FOR STEP 2 - PPI DESIGN

Our example follows the PPIC method illustrated in the Figure 29. Our tool allows the manual addition of all PPI family members using the PPICT graphical notation. To illustrate the use of the tool, we take the example of the PPI Family *Number of Contracts* according to stakeholders' requirements.

The example is based on the PPIs that evaluate the process to *Create a Contract*. To create the PPICT, the root PPI, which designates the PPI family, must first be added. Then, it is possible to add new PPI variants of existing PPIs according to stakeholders' requirements. Each PPI that is added to the tree must respect the rules of the PPICT described in chapter 4.

In our example, the PPICT has 4 levels, cf. Figure 47. At the first level, the root PPI *Number of contracts* can be found. At the second level, 4 associated indicators are illustrated: *Number of Design Contracts*, *Number of Sent Contracts*, *Number of Signed Contracts* and *Number of Active Contracts*.

¹¹ <https://www.adoxx.org/live/meta-modelling-platforms-hierarchy>

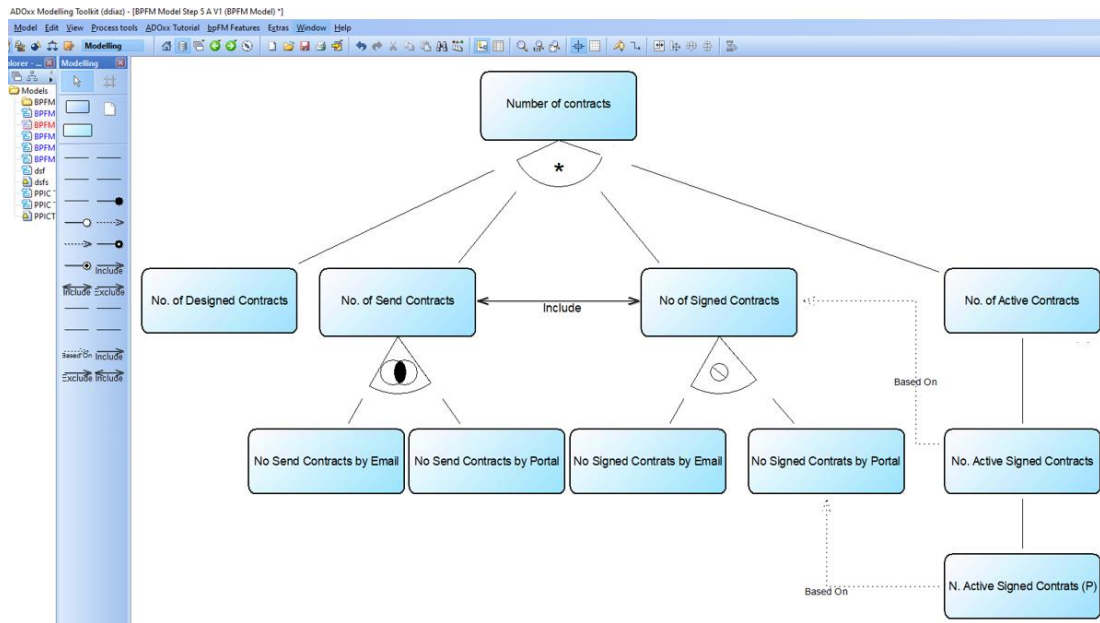


Figure 47 PPICT design of the *Number of Contracts* PPI family

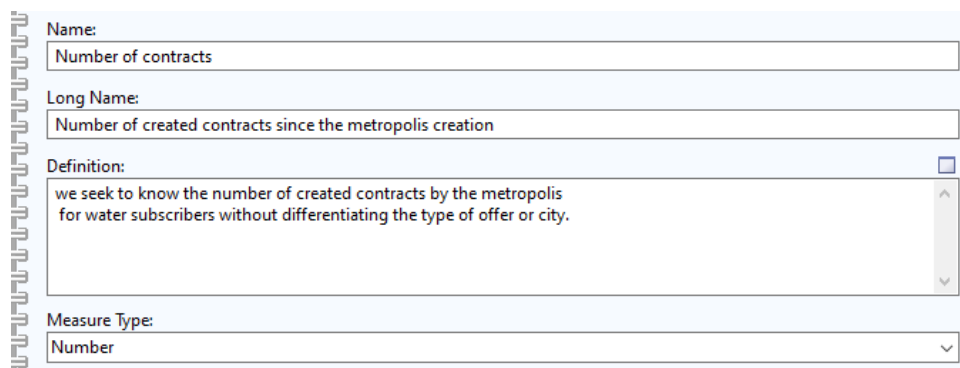
The constraint that joins the root PPI with the PPIs of the second level is a complex constraint since there are PPIs that share common tuples and other PPIs that do not share common tuples, i.e., there are PPIs that share data and other PPIs that do not. Additionally, between the PPI *Number of Sent Contracts* and *Number of Signed Contracts* there is an *Include constraint*; In our example we do not have the *Exclude constraint*.

At the third level there are 5 PPIs, the first two PPIs are variants of the *Number of Sent Contracts*: *Number of Sent Contracts by email* and *Number of Sent Contracts by portal* using an *Overlap constraint*. Also, the following two PPIs are variants of the PPI *Number of Signed Contracts*: *Number of Signed Contracts by Email* and *Number of Signed Contracts by Portal* using a *Disjoint constraint*.

The last PPI of the third level is the *Number of Active Signed Contracts*, which is a variant of the *Number of Active Contracts*, based on the *Number of Signed Contracts*. The notation of “variant of” is represented in the PPICT as *depend on* using a complete line. Likewise, the notation of *based on* is represented using a dotted line with an arrow pointing towards the PPI that is used as the base. Finally, in the fourth and last level of our PPICT example, there is only one PPI *Number of Active Signed Contracts by Portal*, which is a variant of the PPI *Number of Active Signed Contract* and is based on the PPI *Number of Signed Contracts by portal*.

When designing the PPICT, it is necessary to give the essential information of every PPI by double clicking on the desired PPI and editing the description sheet of the window parameters. For instance, the PPI root illustrated in Figure 48 has the following attribute descriptions:

- **The name:** *Number of Contracts.*
- **The long name:** *Number of Created Contracts since the metropolis creation.*
- **The description:** *We seek to know the Number of Created Contracts by the City for water subscribers without differentiating the type of offer or city.*
- **The measure type:** *Number.*



The screenshot shows a form with four fields: 'Name' (Number of contracts), 'Long Name' (Number of created contracts since the metropolis creation), 'Definition' (we seek to know the number of created contracts by the metropolis for water subscribers without differentiating the type of offer or city.), and 'Measure Type' (Number).

Figure 48 Description of PPI *Number of Contracts*

7.2 USE OF THE TOOL FOR STEP 2 - PPI DESIGN

After defining the hierarchy of the PPI family by the competence center, the PPICT tool allows users to define mandatory and optional PPIs as well as the measure representation and the measure aggregation according to stakeholders' definitions and design criteria. This stage must be completed by a competence center, which includes experts of the domains, BP designers and PPI calculation experts. During this step, users must have access to the following attribute definitions to be filled on the PPIC tool:

- I) The *Optional PPI* specifies if a PPI could be optionally configured.
- II) The *Mandatory PPI* specifies that a PPI must be configured by default.

III) The *Measure Representation* in order to visualize PPI's tuples, e.g., value, listing, geographical and chart representation.

IV) The *Measure Aggregation* to aggregate PPI's tuples.

Continuing with our example of PPI *Number of Contracts*, we must define the mandatory PPIs and the optional PPIs. This is done in the first sheet of the parameters of each PPI. For each PPI, the following sheets are available: *description, conditions, linear evaluation, activity-PPI mapping, query* and *configuration*.

In our example, we will only use the *description, conditions, linear evaluation, and query* sheets: the PPI *Number of Contracts* is detailed below:

- I) **PPI type:** *mandatory* since it is the first root PPI. To configure any other PPI of another level, it is necessary to configure the root PPI.
- II) **Measure Aggregation:** *count ()* since the type of measurement is number and we want to know the quantity.
- III) **Measure Representation:** *value* since a quantity of number can only be represented as a value.

Number of contracts (PPI)

PPI Type

Optional

Mandatory

Name:

Number of contracts

Long Name:

Number of created contracts since the metropolis creation

Definition:

we seek to know the number of created contracts by the metropolis for water subscribers without differentiating the type of offer or city.

Measure Type:

Number

Measure Aggregation:

Count

Measure Representation

Value

Listing

Geographical

Chart

Close Reset

Description

Conditions

Linear Evaluation

PPI-Activity Mapping

Query

Configuration

Figure 49 General description of the PPI *Number of Contracts*

In the conditions, users have the possibility to choose to limit the PPI according to a period of time or a deadline. These options are totally independent: if a period is chosen, it is not possible to choose a deadline and vice versa. In our example, no time condition was designated for the PPI *Number of Contracts* cf. Figure 50. In the condition sheet, it is also possible to add entity conditions, for example to filter the PPI with a certain value or values of an attribute of the consulted tables. In the *Number of Contracts* for example, no entity condition is used, cf. Figure 50.

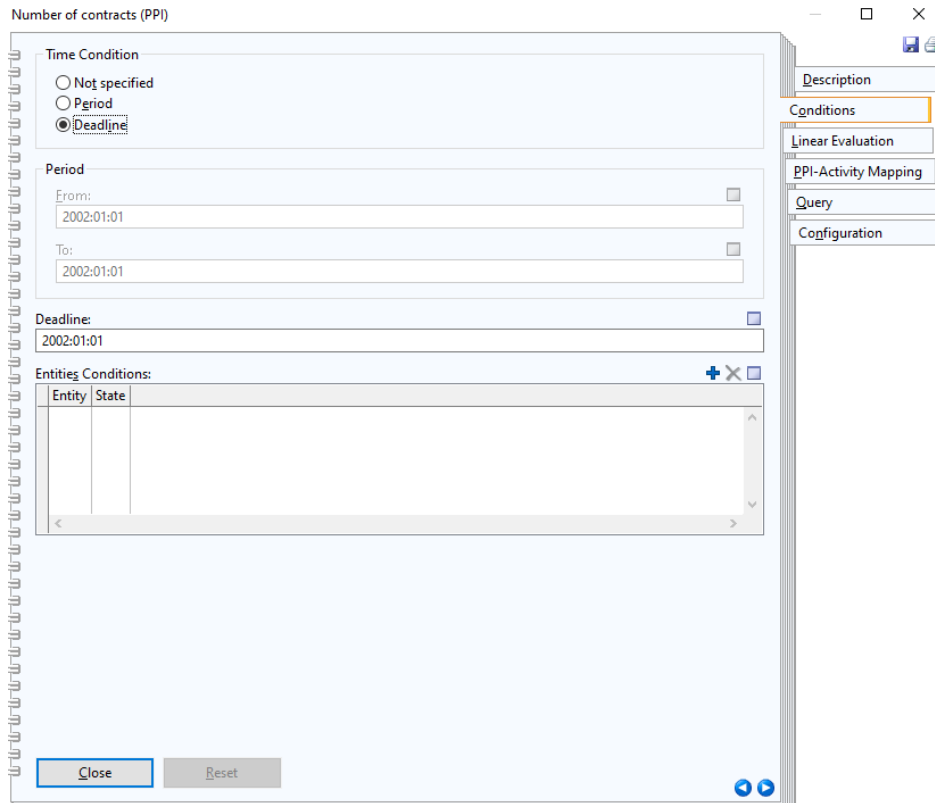


Figure 50 PPI Conditions for the PPI *Number of Contracts*

The linear evaluation described in chapter 4 refers to the estimation of desired values: a minimum, an intermediate and a maximum, cf. Figure 51. In our example, no specific value is chosen for the PPI *Number of Contracts*, since it depends on the execution context. In the case of INCOM, the value of the linear evaluation will depend on the distributor because they do not all manage the same number of subscribers.

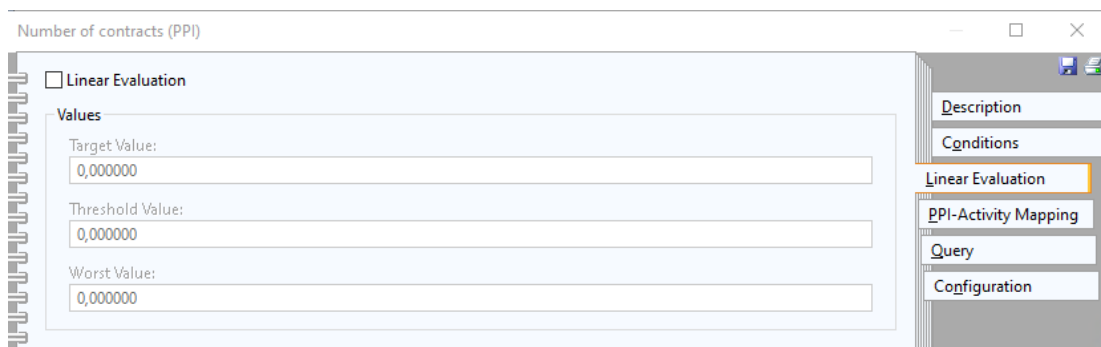


Figure 51 Linear Evaluation: Desired thresholds to evaluation of the PPI results

Up to this point, the PPICT tool can be used independently without having a link with the reference process. The PPICT tool could be used without mapping the process model. It is due to the fact that some software solutions do not have a defined process

model. They enable a global vision of the indicators used to evaluate certain operations or strategies of the company. However, not having a link with the processes implies that it is difficult to determine which activities are evaluated and which tables or attributes in the database are used to calculate a given PPI. We have just a PPI family model to specify the hierarchy of PPIs. Note that a light green square refers to an optional PPI and a dark green square is used for a mandatory PPI, (cf. Figure 52). In our tool, a mandatory PPI is configured by default and an optional PPI is configured manually, so that a red outline is added to the PPIs to represent those PPIs that are configured.

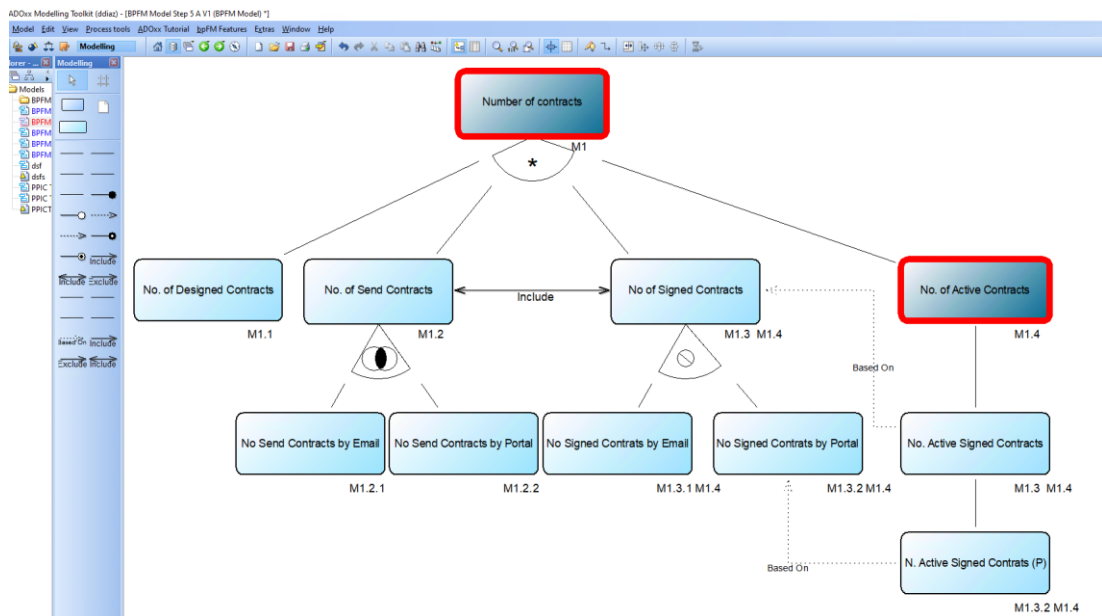


Figure 52 *Number of Contract* PPI family designed the PPIC tool

The goal of step two of the PPIC method is to provide the greatest amount of information possible to calculate the indicators in the next step using an executable language such as SQL. The PPIC tool provides a means to identify this information and make it accessible

7.3 USE OF THE TOOL FOR STEP 3 - BPFM-PPIC ASSOCIATION

The association between a PPI and its respective activities is represented in our tool as a mapping functionality between the PPIC and BPFM models. That is why the BPFM is an input for this step. This association must be done in the parameter sheet of each PPI by listing the associated activities. Each association must have a

specific name: M (mapping) followed by an ascending number that depends on the reference PPI or on the activity to be associated. The association PPI-Activity is based on the PPI hierarchy, e.g., M1 as reference PPI, M1.1 as variant PPI of M1 and M1.1.1 as variant PPI of M1.1. This numerical addition could be done automatically in future versions.

When a variant PPI has neighboring PPIs at the same level, i.e., siblings, it will be given a corresponding number regarding the variant PPI previously added. For instance, M1.1 is the first variant PPI of M1, which means that M1.2 would be the name of the second variant PPI of M1. In this way, the reference PPI can be easily indicated as well as the association. This task must be repeated in the BPFM model, where the associated PPI must be indicated in each activity. This task must be done manually by the user, which is a limit of the current version of the PPIC tool.

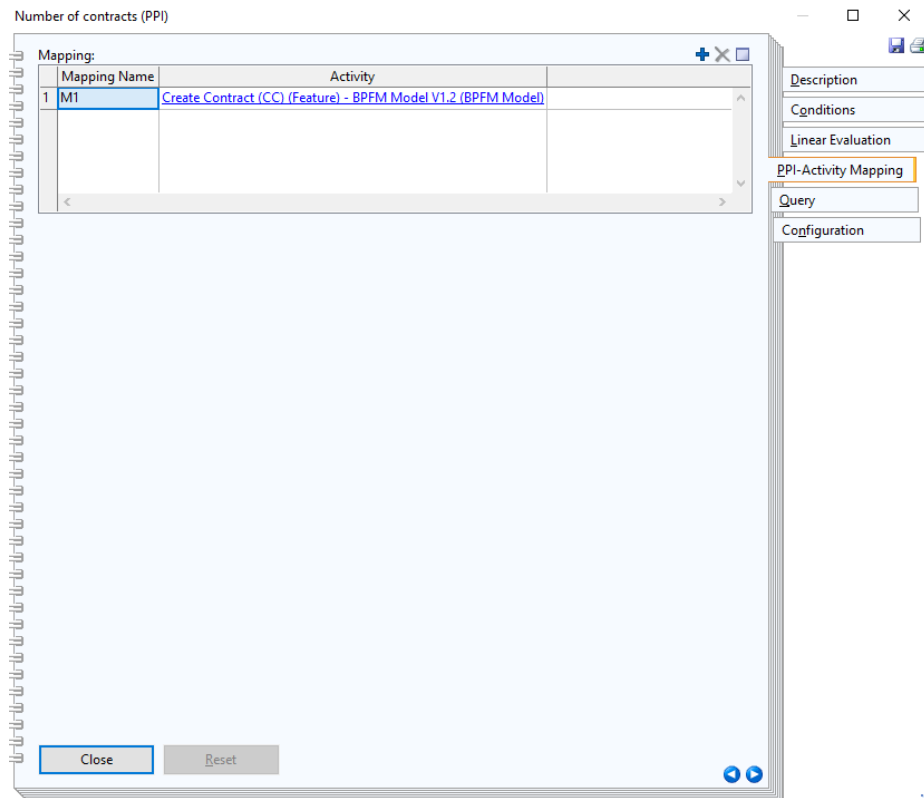


Figure 53 Selection of the activity associated to the PPI *Number of Contracts*

Figure 53 shows an example of a PPI-Activity Mapping parameter sheet. To include an activity associated with a PPI, it is necessary to click on (+) and search for the corresponding activity within the list of available activities. In our example, the PPI *Number of Contracts* is associated with the *Create Contract* activity of the BPFM

Model V1.2 file and the Mapping name is M1. In the process mapping (cf. Figure 54), the user must name the corresponding association and include the tables and applications used to calculate the PPI query. This information can be found on the Process Flow Execution Record as seen in chapter 5.

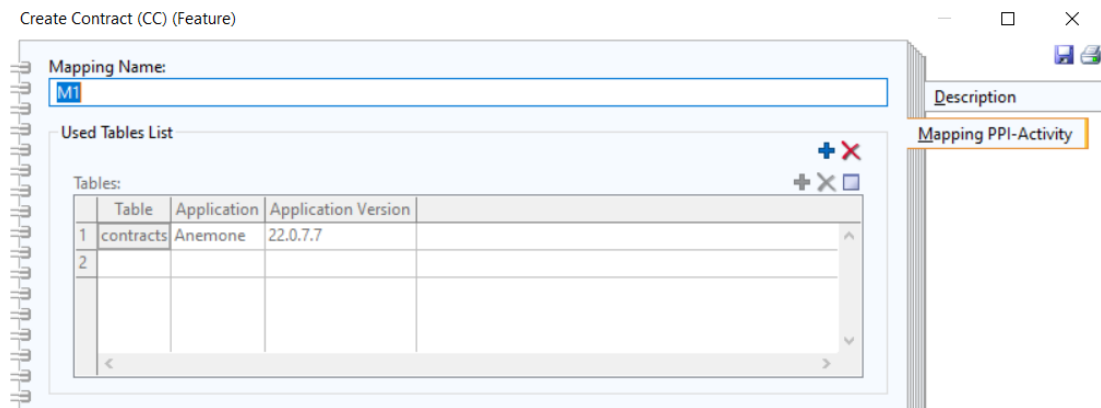


Figure 54 Selection of the associated PPI for the *Create Contract* activity.

The result of this step is shown in Figure 55. It is important to remember that the model of BPFM is a necessary input of our method specifically for this step. In addition to the mapping between the PPI *Number of Contract* and the *Create Contract* activity, all indicators and all activities are associated with each other.

It is important to note that according to the PPIC Metamodel, an activity can be associated with more than one PPI. This is the case of the PPI *Number of Signed Contracts* for instance, which is related to the *Sign Contract* and *Activate Contract* activities. An advantage of the tool is that the associations work as hyperlinks, that is, if the user clicks on M1, for example, it will directly show to the user the related activity *Create Contract*.

At this step in the PPIC Method, if a PPI is automatically configured, it is a mandatory PPI. Currently, in the PPICT tool, if a PPI is configured and the association between PPI-Activity has been made, the tool does not automatically configure the associated activities. This is also a limitation of the PPICT tool that can be improved in future versions.

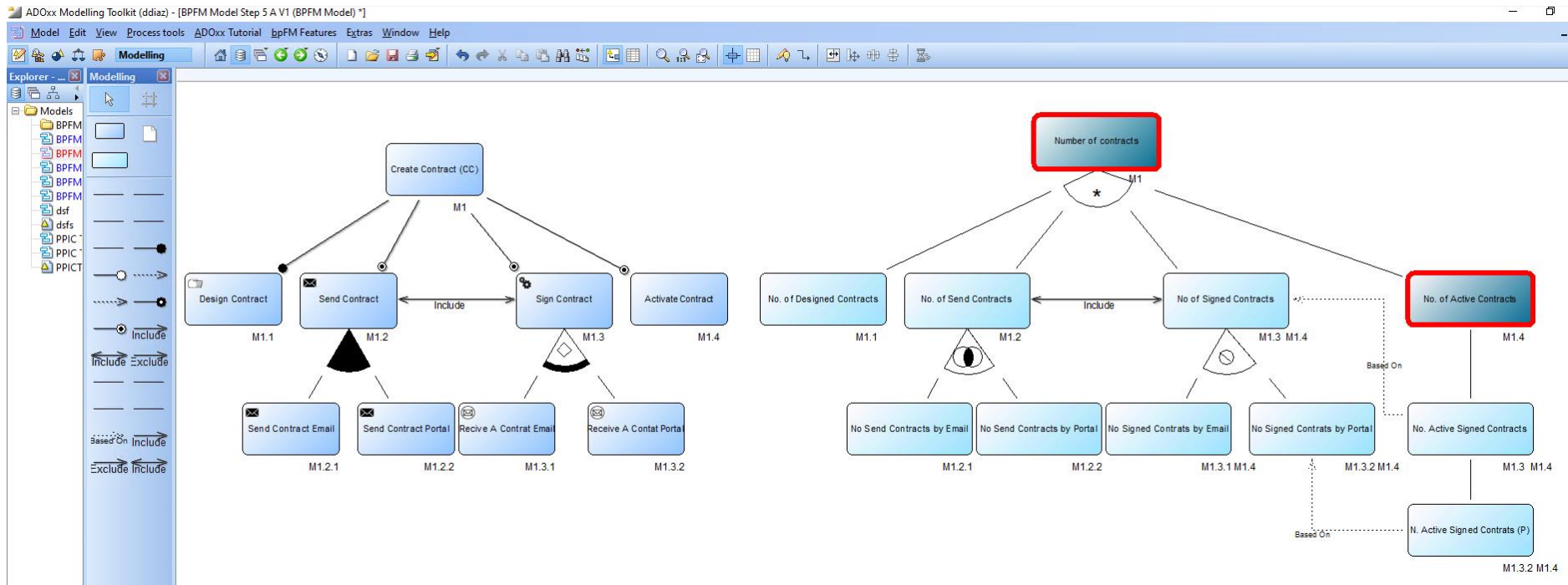


Figure 55 BPFM-PPICT Association

To calculate a PPI, it is necessary to write and execute the associated query. For this, the PPICT tool has an attribute that allows the integration of query executable files. The calculation of a PPI must be done after the association PPI-Activity since it is necessary to know the tables and attributes used by each activity of the process. For instance, in the case of the PPI *Number of Contracts*, a .SQL file was added to the Query Sheet. This file contains the query to calculate and execute the PPI.

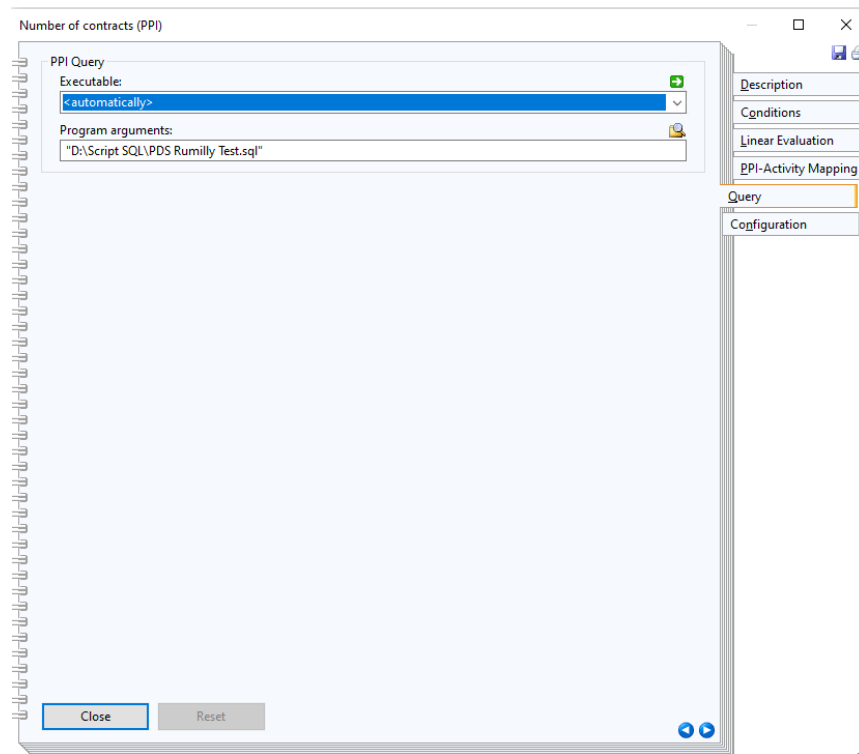


Figure 56 Query association of the PPI *Number of Contracts*

When opening the file, it is possible to execute the associated query. Note that the file must be correctly written and must be executable. To do this, all the tools and drivers necessary for its execution must be installed. Figure 57 shows the SQL request for the PPI *Number of Contracts*, which has an aggregation *count* and a table renamed PDS (Point of Service, Point De Service).

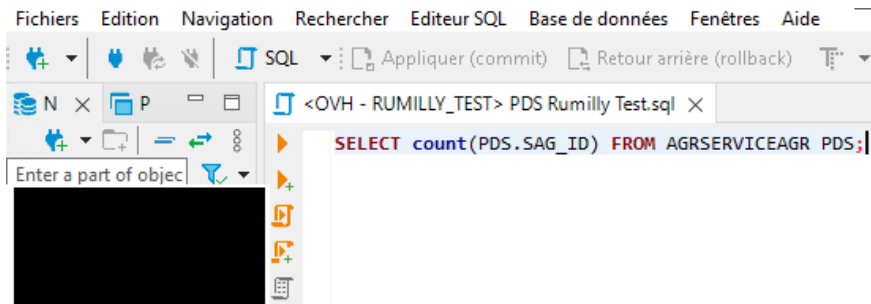


Figure 57 PPI design (SQL execution)

The file execution from indicator calculation tools such as Qlik Sense or Power BI can also be incorporated into the PPICT tool. For instance, Figure 59 shows how the PPI *Number of Active Contracts* can be represented as a value.

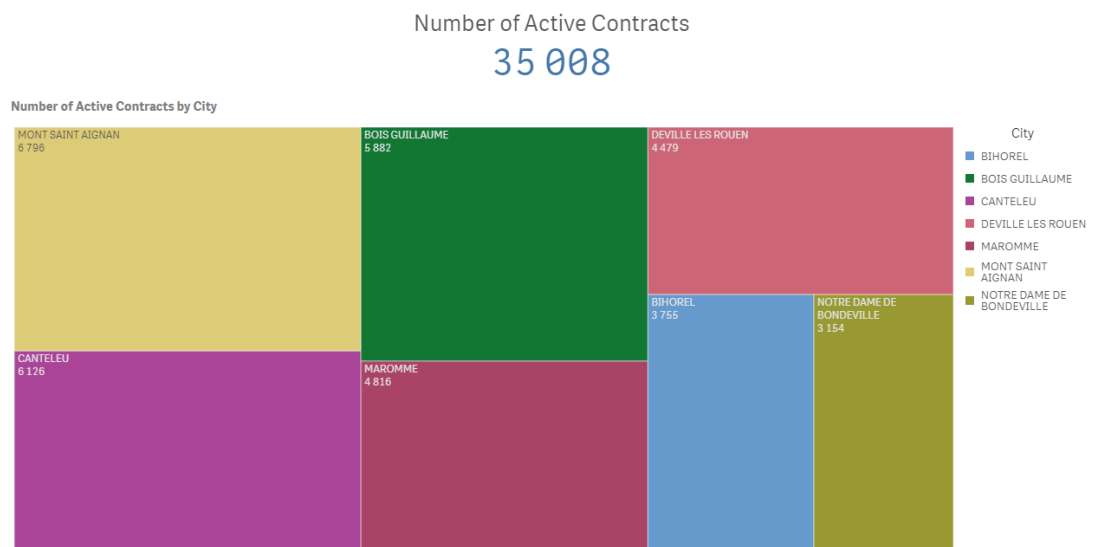


Figure 58 PPI *Number of Active Contracts* and *Number of Active Contracts By City*

Another example of Measure Representation could be a chart of squares. An example of this is the PPI *Number of Active Contracts by City* of Figure 58, where the color and the area of each square depends on the *Number of Active Contracts*. In other cases, this PPI can be represented as a histogram for example in the case of an *E-contract*, i.e., *electronic contract*, which is a *contract signed by the portal* (cf. Figure 59). It is important to know that the business intelligence systems as Qlik Sense or Power BI allow us to extract data at a given moment, or sometimes in real time. In our case, all PPIs are calculated daily using the Qlik Sense data load tools.

Number of Active Contracts Signed By Portal

56 500

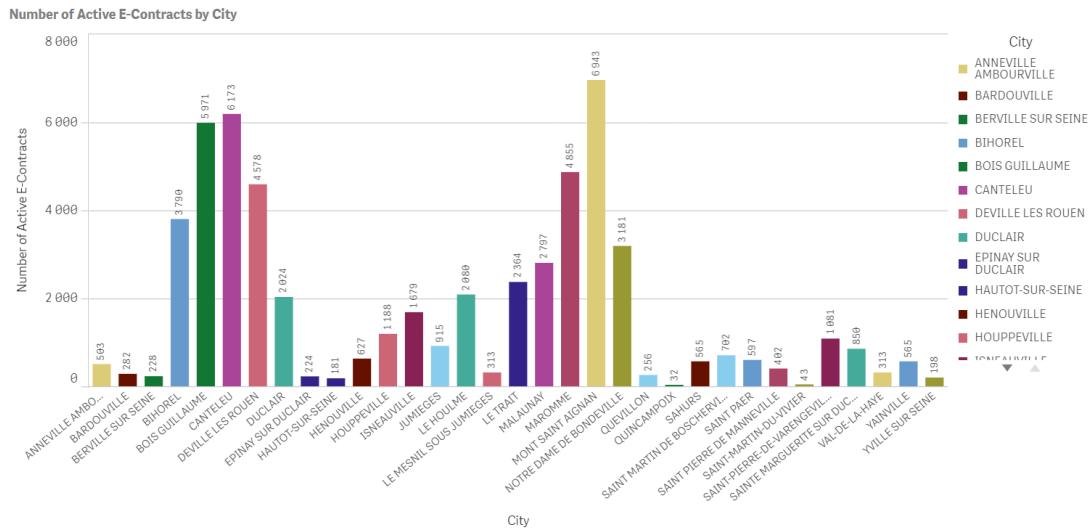


Figure 59 PPI *Number of Active Contracts Signed by Portal* and *Number of Active Contracts by City*

7.4 USE OF THE TOOL FOR STEP 4 -PPICT CONFIGURATION

In this step, the user must select optional PPIs to be deployed using the configuration sheet of the parameter window. By default, all the optional PPIs are not configured, and all the mandatory PPIs are configured automatically. For this step, it is important that all the PPIs that are configured have an executable query, since the objective is to evaluate the process variant to be deployed.

For example, in Figure 60, the PPI *Number of Designed Contracts* is not selected by default in the case of a mandatory PPI. Users could select an optional PPI through the parameter window. The result of this step is shown in Figure 61 where there are three PPIs configured: the two mandatory PPIs *Number of Contracts* and *Number of Active Contracts* and the optional PPI configured *Number of Designed Contracts*.

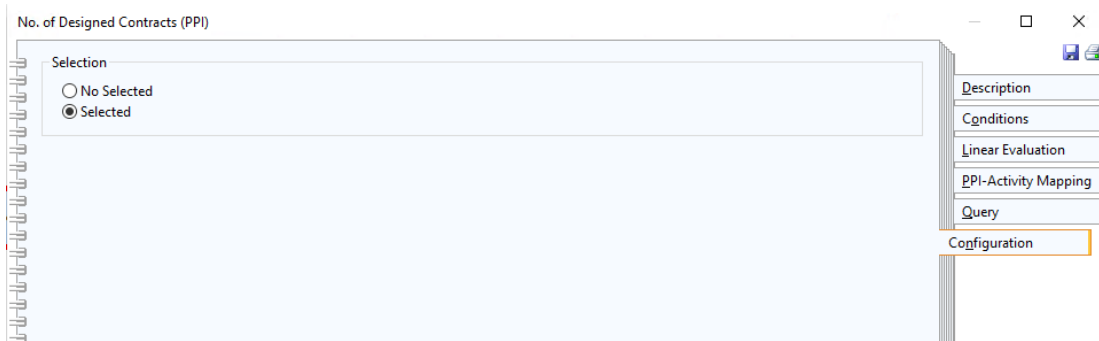


Figure 60 PPI Configuration sheet on our PPIC tool

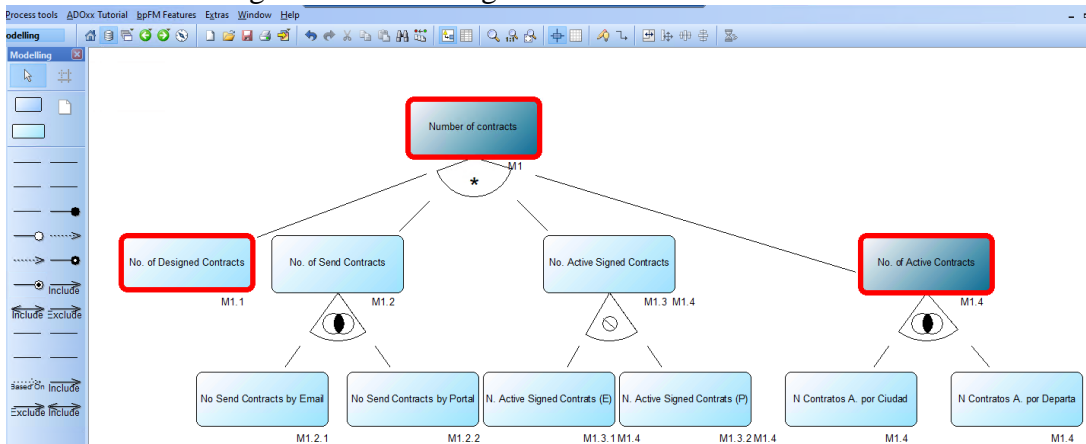


Figure 61 PPICT Configuration on our PPIC tool

7.5 USE OF THE TOOL FOR STEP 5 - PPICT CONFIGURATION CHECKING

In this step, configuration alignment errors are verified. A configuration error may occur if there are PPIs configured for which the corresponding activities are not configured. For the moment, this verification must be done manually. We are aware that it is a limitation of our PPIC tool. The automatic verification task to establish if there are configured indicators without associated configured activities can be executed through the PPI-Activity mapping.

When there is a misalignment between the configurations of the PPICT and the BPFM model, the user must go back to the mapping sheet of each activity and/or PPI studied to change the configuration. This task must be performed as many times as necessary to align the configuration between the BPFM model and the PPICT. Special care should be taken with dependencies between PPIs themselves and between activities themselves, e.g., inclusion or exclusion. These dependencies can influence the modeling and the desired configuration of the two models.

For example, in Figure 62, the PPIs *Number of Contracts*, *Number of Active Contract* and *Number of Designed Contracts* are configured. However, only the PPIs *Number of Contracts* and *Number of Designed Contracts* are mapped to the corresponding **configured** activities, i.e., those PPIs are mapped to the *Create Contract activity* and to the *Design Contract activity*. The PPI *Number of Active Contracts*, is mapped to the **non-configured** *Activate Contract activity*. In this case the user must configure *Activate Contract activity* in the BPFM model to align both families, the process family, and the PPI family.

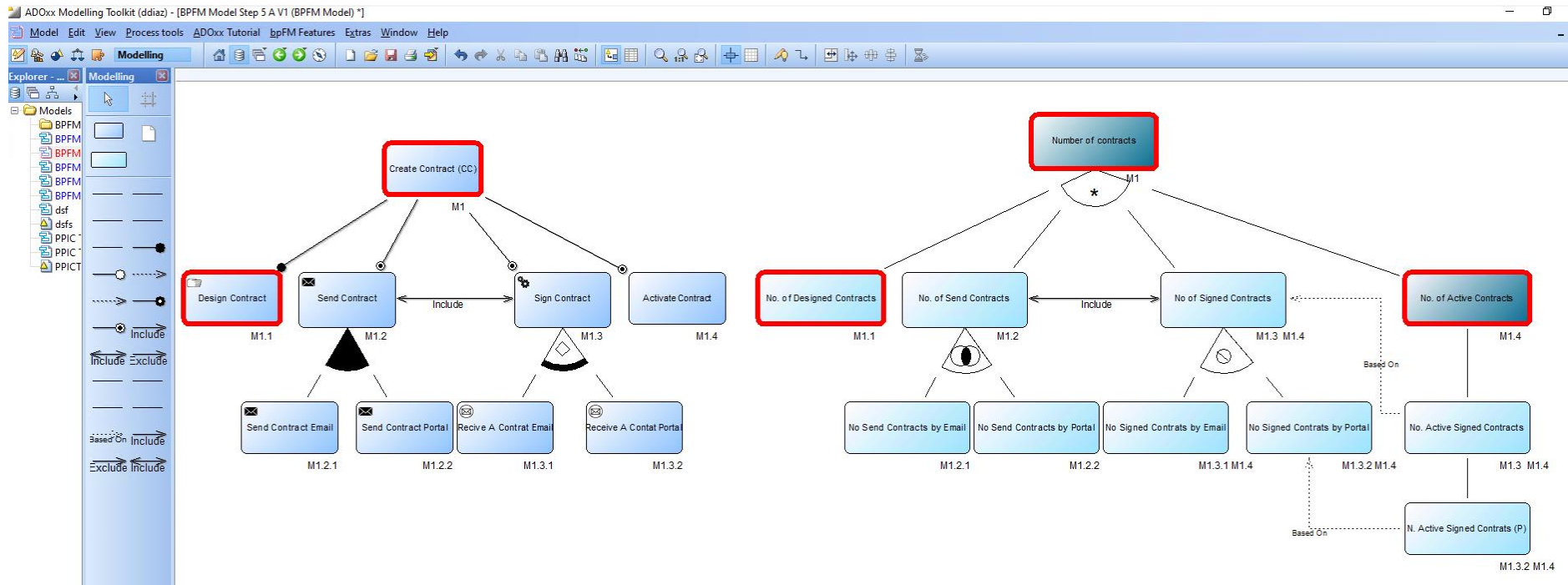


Figure 62 Mismatching between BPFM (*Activate Contract*)-PPIC (*Number of Active Contracts*) a) BPFM Configuration, b) PPIC configuration.

Once the mapping error has been corrected and the *Activate Contract* activity configured, the BPFM-PPIC association is correct. It can be seen in Figure 64 that there are three configured activities as well as three equally configured PPIs. As explained in chapter 4, more PPIs can be configured than activities, since an activity can be evaluated through different indicators. However, PPIs will always need at least one associated activity to be calculated, in order to obtain the associated data after the process execution.

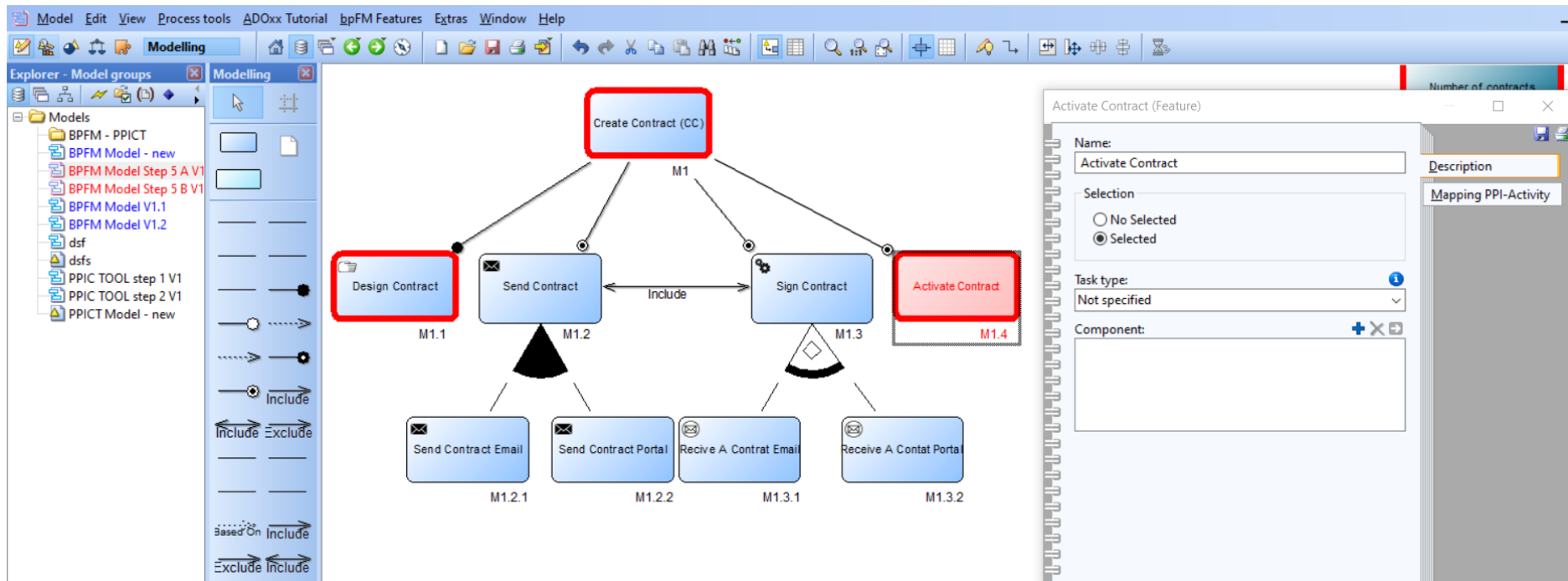


Figure 63 Configuration of the *Active Contract* Activity in the BPFM model

To align the BPFM model and the PPICT, the *Active Contract* activity has been configured. This is achieved in the same way as the configuration of a PPI, in the parameter window by selecting the corresponding activity.

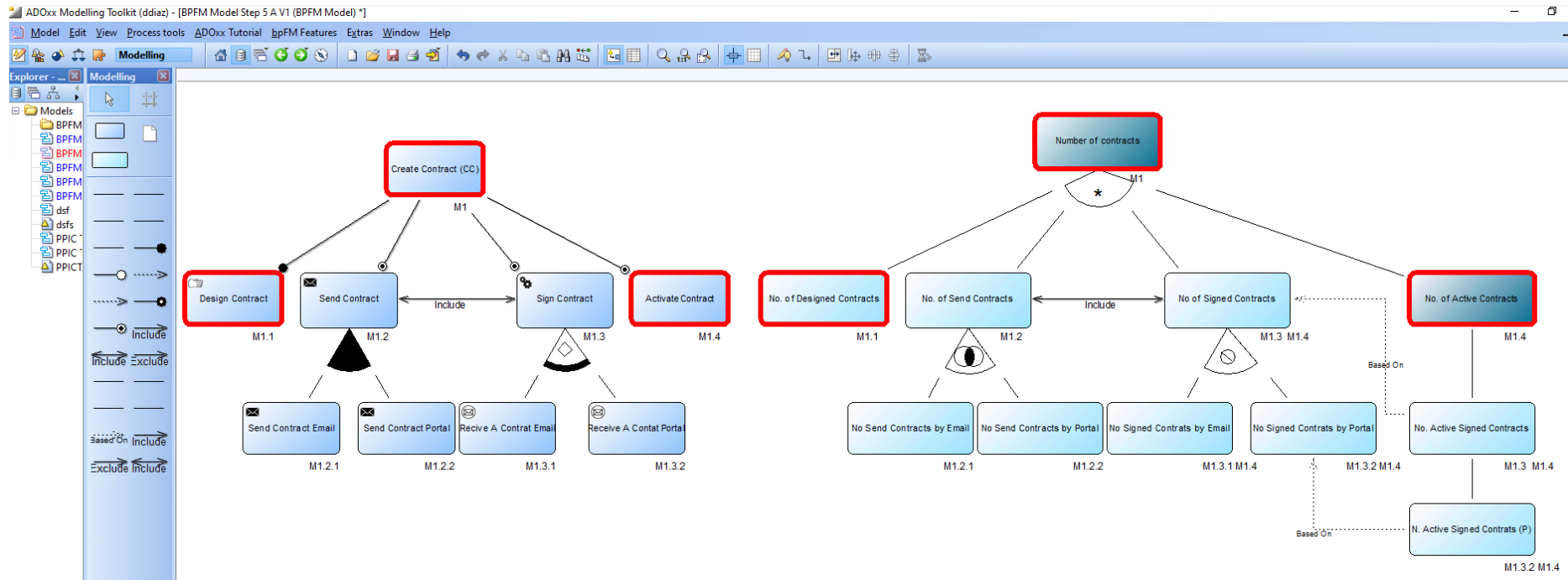


Figure 64 Correct BPFM-PPIC mapping a) BPFM Configuration, b) PPICT configuration

In summary, the PPIC tool allows the modeling of a *PPI Family* following the steps of the PPIC method. The PPIC tool provides the possibility of linking the activities of a reference process to the PPIs to evaluate the process performance. Some automatic validations included in the PPIC tool are the configuration a mandatory indicator by default, alerting that neither the associated activities nor the reference PPI are configured.

7.6 CONCLUSION AND LIMITATIONS

The PPIC tool is designed to be run on the Windows operating system as an executable file (.exe). Our goal is to contribute to the development of research: developed on the ADOxx platform, the PPIC tool can be extended and improved by the scientific community because the platform is open-source.

A limitation of the PPICT tool is that the user must access the PPI attributes to visualize and execute the query written in an external tool, e.g., Qlik Sense, Power BI, etc. Thus, it is necessary to have a minimum knowledge of Business Intelligence tools.

Another limitation is that when the *family of PPIs* is too large, the position of a new PPI is without doubt difficult because there may be several potential reference PPI. For this reason, it could be useful to propose a potential reference PPI.

Another limitation appears when we add new joins. This task is difficult for users who have little prior knowledge of the data and have no access to the database documentation. Currently, when users make a mistake with the choice of the constraints, they cannot verify if the variant PPIs are really the variant PPIs of a reference PPI. Neither can they check if the containment condition of a variant PPI is respected or if the relational constraints are respected such as the Overlap Constraint or the Disjoint Constraint. This is why it is required to have at least basic knowledge of data base modeling.

Chapter 8: Conclusions, limitations, and perspectives

As we have seen in the document, our different contributions aim at integrating the process performance indicator (PPI) Variability in the context of Customizable Process Models. Our main goal is to capture and model multiple variants of a PPI linked to multiple variants of a business process, in a consolidated manner. This chapter concludes the documents and presents some limitations and perspectives of our work, providing a summary of the propositions and possible improvements of the PPIC Method, the PPICT and our software tool.

8.1 CONCLUSION

Customizable Process Models and the corresponding PPIs are usually modeled separately. Since the support of Process Variability complicates the PPI definitions and calculations, modeling PPI variants with no explicit link with the related Customizable Process generates redundant models, making adjustment and maintenance difficult. Relying on our experience as software publishers for public administrations, this Ph.D. thesis proposes the PPICT approach. We extend the Business Process Feature Model (BPFM) approach by integrating PPI Variability to facilitate the creation and modeling of PPIs in the context of Customizable Process Models. We presented the Process Performance Indicator Calculation Tree (PPICT) to model the PPI Variability and the relation between PPIs. The PPICT can be linked to the Customizable Process Models relying on the BPFM approach to identify the activities to evaluate and thus answer our second research question. We also presented the PPIC method, which details the process to integrate the PPI modeling to the Customizable Process Models and thus answers our first research question. Our contribution lies in three main axes:

- I) A method composed 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.

III) A software tool supporting the method developed on ADOxx.

The PPIC method is illustrated in a real utility distributor case and has been validated by PPI developing experts and novices. The validation was carried out through a user-centered evaluation, which allows the use of the PPICT to model a PPI family linked to a BP family. Furthermore, the PPIC method complements the related works of Customizable Process Models by broadening the process variants to be measured and the measures themselves through the PPI calculation and variability modeling, thus answering our third research question.

The PPIC Method, which includes the construction and use of the PPICT has been implemented in a software tool, which allows the executions of SQL queries to calculate PPIs. The tool integrates query execution and business intelligence systems to implement all PPI family members. This tool supports the definition and modeling of PPI references and their corresponding variants, decreasing design-time and calculation-time of PPIs linked to Customizable Process Models.

In organizations such as INCOM, software publishers need to carry out mechanisms to enable the evaluation and continuous improvement of Customizable Process Models. PPICT proposes a mechanism to model PPIs, which are quantifiable, measurable metrics from data generated during the process. Nevertheless, the PPI management cannot be restricted only to business process evaluation. It must be extended throughout the entire business process management lifecycle. That is why we proposed the PPICT method to secure close bond between Customizable Process Model design, execution and improvement with PPI design, execution, and improvement. Thus, we make sure that the PPI management lifecycle is integrated to the Customizable Process Model management lifecycle.

Our contributions are based on a mapping and a wide analysis of PPIs and customizable processes used by different utility distributors. The PPIC method formalizes and exploits the links between process families and PPI families. The benefits of this union could be measured in terms of reduced design-time to calculate a new variant PPI.

In the utility distribution context, some PPIs are recurrent and must be deployed for all utility distributors. This happens because they must present mandatory declarations to regulatory entities. However, it does not mean that the business

processes deployed are the same nor that the PPIs are identically calculated. Today, whenever possible, INCOM deploys the same process variants to avoid re-designing processes or re-calculating PPIs. This, in other to standardize the business processes and PPIs. Unfortunately, this implies that the utility distributor must adapt themselves to the deployed processes.

Currently, the PPICT tool allows the modeling and calculation of INCOM PPIs. This guarantees that employees can use the PPIC method to model PPI families in the context of Customizable Process Models.

8.2 LIMITATIONS

PPICT Limitations

A limitation of the PPICT, highlighted in the empirical evaluation, is that there is no view of the query on a single diagram. Likewise, another limitation concerns mistakes that users can make in their choice of constraints if they do not know the theory of sets and if they do not the data to be analyzed.

Furthermore, the complexity of adding a new relation when the user has little prior knowledge of the data to be analyzed. The terminology used is sometimes too complicated for users that are unfamiliar with process families. This task is also difficult for users who have little prior knowledge of the data and have no access to the database documentation. Currently, when users make a mistake with the choice of the constraints, they cannot verify the correctness of constraints and conditions. For this, it is important to have a minimum of knowledge of data modeling.

PPIC Method limitations

A limitation of the PPIC Method is that when the *family of PPIs* is too large, the position of a new PPI is difficult to determine because there may be several potential Reference PPIs. For this, it could be useful to propose a potential reference PPI in the PPIC tool.

Another limitation of the PPIC Method is that we did not model all possible variants of all deployed BPs of INCOM's clients. There are more than 150 process references with more than 7,500 deployed process variants. We have made a cartography of the two most common variable processes. To use and exploit the PPIC

method, it is recommended to cartography not only the variable processes but all possible process variants of those processes.

PPIC tool limitations

The number of utility distributors' business activities is gaining increasing importance. Currently, it is difficult to know the maximum number of PPIs and activities that our tool can support. However, we are sure that the more PPIs there are in the PPICT, the more complicated it is to find the corresponding reference PPI and the level at which each PPI should be located.

Regarding the limitations due to the PPIC tool, the automatic detection of a reference PPI and automatic suggestion of variant PPI should be mentioned.

Also, the user must access the PPI attributes to visualize and execute the query written in an external tool, e.g., Qlik Sense, Power BI, etc. Thus, it is necessary to have a minimum knowledge of Business Intelligence tools.

The PPIC tool does not allow to automatically propose the possible reference PPIs based on the name of the new PPI and the operators used by existing PPIs. For instance, if the PPI that we want to add is the *Number of Signed Contracts by Portal*, the reference PPI is the *Number of Signed Contracts*.

PPICT and PPIC Method limitation

Our contribution does not aim to model and calculate PPIs in the context of non-functional properties such as Service Level Agreement (SLA). SLA refers to the contractual terms and agreements governing the duration of the service engagement between the service provider and service consumer. The PPICT approach could design a PPI family to provide a global overview of SLAs (Service Level Agreement) as long as there is a business process that generates the corresponding data. As a modeling tool of Customizable Process Models and a real PPI execution tool, the PPICT can be interconnected to different databases.

PPICT and PPIC tool limitations

A limitation is that our model and tool do not support PPI modeling based on NoSQL databases.

8.3 PERSPECTIVES

8.3.1 Short-term perspectives

Our main perspective would be the modeling of the PPI Variability in business process families that use non-relational storage systems, e.g., by extending the PPICT notation, the links between PPIs and by ensuring that the data is data-model agnostic. Another extension of our contribution could be to automatically identify the possible variant PPIs, e.g., by identifying a similar PPI name or by identifying the operators and fields used by other PPIs. The possible extension is an open research topic that could be explored in our future research. (Diaz, 2020; Diaz et al., 2019, 2021)

8.3.1 Medium-term perspectives

Another improvement would be to propose a potential reference PPI for a new variant PPI. Likewise, when users choose a PPICT constraint such as the Overlap Constraint or the Disjoint Constraint, it could be useful to automatically verify if the variant PPIs are really the variant of the selected reference PPI. Or if the inclusion rule of a subset of tuples between a variant and reference PPI is respected. It would also be interesting to check if the relational constraints are respected.

8.3.1 Long-term perspectives

The integration of energy-related metrics referred to as Green Performance Indicators (GPIs) is a valuable perspective in which the approach could be extended. Monitoring the level of consumed energy by every process activity (Del-Río-Ortega et al., 2019) and evaluating variants relying on GPIs could favor greener Business Process Models. Our PPICT metamodel could be extended to support the GPIs in the context of Customizable Process Models.

We have presented several concepts to define and model PPI Variability. However, we do not include the feedback obtained during the process execution because our solution does not use a business process model system. That is why it could be interesting to integrate an automatic post execution analysis to determine the advantages and disadvantages of the Customizable Process Model evaluations using the PPICT, which could be monitored through process mining techniques.

Bibliography

- Aalst, W. van der, Lohmann, N., Rosa, M. La, & Xu, J. (2010). Correctness ensuring process configuration: An approach based on partner synthesis. *International Conference on Business Process Management*, 95–111.
- Andrews, T., Curbera, F., Dholakia, H., Golland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., & Thatte, S. (2003). *Business process execution language for web services*. version.
- Cognini, R., Corradini, F., Polini, A., & Re, B. (2016). Business Process Feature Model: An Approach to Deal with Variability of Business Processes. *Domain-Specific Conceptual Modeling, July*, 171–194. <https://doi.org/10.1007/978-3-319-39417-6>
- Davis, R., & Brabander, E. (2007). *ARIS design platform: getting started with BPM*. Springer Science & Business Media.
- Dechow, D., Stoltz, P., Amundson, J., & Spentzouris, P. (2005). Synergia: An Advanced Object-Oriented Framework for Beam Dynamics Simulation. *Proceedings of the 2005 Particle Accelerator Conference*, 1925–1927.
- Delgado, A., Weber, B., Ruiz, F., de Guzmán, I. G.-R., & Piattini, M. (2014). An integrated approach based on execution measures for the continuous improvement of business processes realized by services. *Information and Software Technology*, 56(2), 134–162.
- Del-Río-Ortega, A., Resinas, M., Cabanillas, C., & Ruiz-Cortés, A. (2013). On the definition and design-time analysis of process performance indicators. *Information Systems*, 38(4), 470–490.
- Del-Río-Ortega, A., Resinas, M., Durán, A., Bernárdez, B., Ruiz-Cortés, A., & Toro, M. (2019). Visual PPINOT: A graphical notation for process performance indicators. *Business & Information Systems Engineering*, 61(2), 137–161.
- Del-Río-Ortega, A., Resinas, M., Durán, A., & Ruiz-Cortés, A. (2016). Using templates and linguistic patterns to define process performance indicators. *Enterprise Information Systems*, 10(2), 159–192.
- Diaz, D. (2020). Integrating PPI Variability in the Context of Customizable Processes by Extending the Business Process Feature Model. *2020 IEEE 24th International Enterprise Distributed Object Computing Workshop (EDOCW)*, 80–85.
- Diaz, D., Cortes-Cornax, M., Labbé, C., & Faure, D. (2019). Modélisation de la variabilité des indicateurs dans le cadre des administrations de services publics. *INFORSID (INFormatique Des ORganisations et Systèmes d'Information et de Décision)*.
- Diaz, D., Cortes-Cornax, M., Labbe, C., & Faure, D. (2021). A Method for Modeling Process Performance Indicators Variability Integrated to Customizable Processes Models. *International Conference on Research Challenges in Information Science*, 72–87.
- Döhring, M., & Zimmermann, B. (2011). vBPMN: event-aware workflow variants by weaving BPMN2 and business rules. In *Enterprise, Business-Process and Information Systems Modeling* (pp. 332–341). Springer.
- Estrada Torres, B., Torres, V., Río Ortega, A. del, Resinas Arias de Reyna, M., Pelechano, V., & Ruiz Cortés, A. (2016). Defining PPIs for Process Variants based on Change Patterns. *JCIS 2016: XII Jornadas de Ciencia e Ingeniería de Servicios (2016)*,.

- Estrada-Torres, B., Del-Río-Ortega, A., Resinas, M., & Ruiz-Cortés, A. (2016). Identifying variability in process performance indicators. *International Conference on Business Process Management*, 91–107.
- Estrada-Torres, B., Del-Río-Ortega, A., Resinas, M., & Ruiz-Cortés, A. (2021). Modeling Variability in the Performance Perspective of Business Processes. *IEEE Access*, 9, 111683–111703.
- Friedenstab, J.-P., Janiesch, C., Matzner, M., & Muller, O. (2012). Extending BPMN for business activity monitoring. *2012 45th Hawaii International Conference on System Sciences*, 4158–4167.
- Glaser, B. G., & Strauss, A. L. (1967). *The discovery of grounded theory (Rev. ed., 1999 ed.)*. Chicago: Aldine.
- Golfarelli, M., Rizzi, S., & Cella, I. (2004). Beyond data warehousing: what's next in business intelligence? *Proceedings of the 7th ACM International Workshop on Data Warehousing and OLAP*, 1–6.
- Gottschalk, F., Van Der Aalst, W. M. P., Jansen-Vullers, M. H., & La Rosa, M. (2008). Configurable workflow models. *International Journal of Cooperative Information Systems*, 17(02), 177–221.
- Gottschalk, F., Wagemakers, T. A. C., Jansen-Vullers, M. H., van der Aalst, W. M. P., & La Rosa, M. (2009). Configurable process models: Experiences from a municipality case study. *International Conference on Advanced Information Systems Engineering*, 486–500.
- Hallerbach, A., Bauer, T., & Reichert, M. (2010). Capturing variability in business process models: the Provop approach. *Journal of Software Maintenance and Evolution: Research and Practice*, 22(6-7), 519–546.
- Kaplan, R. S., Norton, D. P., Kaplan, R. S., & Norton, D. P. (1992). The Balanced Scorecard – Measures that Drive Performance The Balanced Scorecard — Measures. *Harvard Business Review*.
- Kumar, A., & Yao, W. (2009). Process materialization using templates and rules to design flexible process models. *International Workshop on Rules and Rule Markup Languages for the Semantic Web*, 122–136.
- La Rosa, M., Dumas, M., Ter Hofstede, A. H. M., & Mendling, J. (2011). Configurable multi-perspective business process models. *Information Systems*, 36(2), 313–340.
- La Rosa, M., & Gottschalk, F. (2009). Synergia-Comprehensive Tool Support for Configurable Process Models. *BPM (Demos)*, 489.
- La Rosa, M., Van Der Aalst, W. M. P., Dumas, M., & Milani, F. P. (2017). Business process variability modeling: A survey. *ACM Computing Surveys (CSUR)*, 50(1), 2.
- Lakens, D. (2022). Sample size justification. *Collabra: Psychology*, 8(1), 33267.
- Landström, A., Almström, P., Winroth, M., Andersson, C., Öberg, A. E., Kurdve, M., Shahbazi, S., Wiktorsson, M., Windmark, C., & Zackrisson, M. (2018). A life cycle approach to business performance measurement systems. *Procedia Manufacturing*, 25, 126–133.
- Li, S.-T., Shue, L.-Y., & Lee, S.-F. (2008). Business intelligence approach to supporting strategy-making of ISP service management. *Expert Systems with Applications*, 35(3), 739–754.
- Lönn, C.-M., Uppström, E., Wohed, P., & Juell-Skielse, G. (2012). Configurable process models for the Swedish public sector. *International Conference on Advanced Information Systems Engineering*, 190–205.

- Mandran, N., & Dupuy-Chessa, S. (2018). Supporting experimental methods in information system research. *2018 12th International Conference on Research Challenges in Information Science (RCIS)*, 1–12.
- Marr, B. (2012). *Key Performance Indicators (KPI): The 75 measures every manager needs to know*. Pearson UK.
- Marshall, B., Cardon, P., Poddar, A., & Fontenot, R. (2013). Does sample size matter in qualitative research?: A review of qualitative interviews in IS research. *Journal of Computer Information Systems*, 54(1), 11–22.
- Negash, S., & Gray, P. (2008). Business intelligence. In *Handbook on decision support systems 2* (pp. 175–193). Springer.
- Ngo, V. M., Le-Khac, N.-A., & Kechadi, M. (2019). Designing and implementing data warehouse for agricultural big data. *International Conference on Big Data*, 1–17.
- Ognjanovic, I., Mohabbati, B., Gaevic, D., Bagheri, E., & Bokovic, M. (2012). A metaheuristic approach for the configuration of business process families. *Services Computing (SCC), 2012 IEEE Ninth International Conference On*, 25–32.
- OMG. (2013). *Business Process Model and Notation (BPMN) . 2.1*(December).
- OMG. (2016). *Case Management Model and Notation (CMMN) . 1.1*(December).
- Parmenter, D. (2015). *Key performance indicators: developing, implementing, and using winning KPIs*. John Wiley & Sons.
- Popova, V., & Sharpanskykh, A. (2010). Modeling organizational performance indicators. *Information Systems*, 35(4), 505–527.
- Rahim, A. G., Ofuani, A. B., & Olonode, O. P. (2018). *Trends in business performance measurement: A literature analysis*.
- Reichert, M., Hallerbach, A., & Bauer, T. (2015). Lifecycle management of business process variants. In *Handbook on Business Process Management 1* (pp. 251–278). Springer.
- Rodriguez, R. R., Saiz, J. J. A., & Bas, A. O. (2009). Quantitative relationships between key performance indicators for supporting decision-making processes. *Computers in Industry*, 60(2), 104–113.
- Rosenberg, A., Chase, G., Omar, R., Taylor, J., & von Rosing, M. (2011). *Applying real-world BPM in an SAP environment*. Galileo Press Bonn, Bosten.
- Saldivar, J., Vairetti, C., Rodríguez, C., Daniel, F., Casati, F., & Alarcón, R. (2016). Analysis and improvement of business process models using spreadsheets. *Information Systems*, 57, 1–19.
- Schnieders, A., & Puhlmann, F. (2006). Variability Mechanisms in E-Business Process Families. *Proceedings of the 9th International Conference on Business Information Systems (BIS'06)*, 85, 583–601.
- Shigarov, A. O. (2015). Table understanding using a rule engine. *Expert Systems with Applications*, 42(2), 929–937.
- Van der Aa, H., Del-Río-Ortega, A., Resinas, M., Leopold, H., Ruiz-Cortés, A., Mendling, J., & Reijers, H. A. (2016). Narrowing the business-IT gap in process performance measurement. *International Conference on Advanced Information Systems Engineering*, 543–557.
- Velu, A. (2021). Influence of business intelligence and analytics on business value. *International Engineering Journal For Research & Development*, 6(1), 9–19.

Appendices

Appendix A

Experimentation interview guide for the PPICT method

Description	Interview guide.
Goal	This document contains the questions to ask the experts during the experiment corresponding to the PPICT.
Diffusion	LIG and INCOM.
Dated	03/05/2021
Author	Diego DIAZ
Reference Version	1.4
Participant	

Version history			
Version	Dated	Author	Description
1.0	02/28/2020	Diego DIAZ	Document creation
1.1	03/02/2020	Mario CORTES CORNA X	Modification of Q3 and Q4
1.2	02/03/2020	Diego DIAZ	Update
1.3	03/03/2020	Nadine MANDRAN	Updated questions for an interview guide
1.4	05/03/2021	Diego DIAZ	Update

Q2: Context of the experiment

Interview: Diego takes notes and records

I have just introduced you to the notion of variability of indicators,

- What do you think of this notion of variability of indicators?

- Have you ever encountered this problem of indicator variability?

- What solutions did you use to address this problem?

- In your opinion, how are the indicators impacted by process variability?

Have the participant fill in pages 4 to 7 and then discuss with him that he has proposed.

Q3: PPICT Concepts

We will ask you to rate from 1 to 4 the clarity of the definitions related to the PPCIT model. If the definition is not clear enough, you can modify it in the comment area.

Clarity: 1 not at all clear... 4 quite clear.

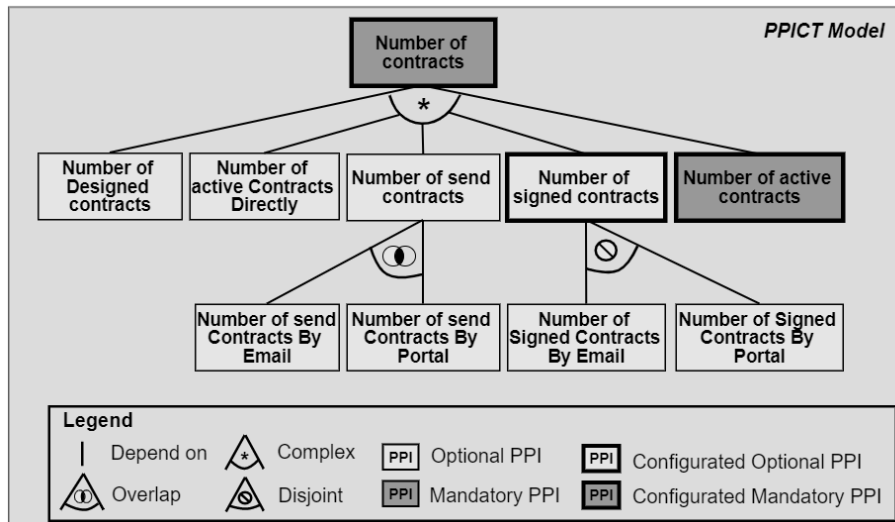
Concept	Definition	Clarity
Indicator	A strategic and operational measure that can be calculated differently depending on the processes deployed.	1 2 3 4
	Comments:	
Reference PPI	Indicator which serves as a basis for the calculation of its variants in the same family of indicators.	1 2 3 4
	Comments:	
Variant PPI	Indicator derived from a benchmark indicator from the same family of indicators. All tuples of a variant PPI before applying any aggregation or any group by are subsets smaller or equal to all tuples of its reference PPI before applying any aggregation or any group by. A non-leaf variant indicator can also be a benchmark indicator. In addition, a variant indicator only adds operators to the query structure of its reference indicator. That is, a variant indicator does not remove any operator from its reference indicator.	1 2 3 4
	Comments:	
Calculation type	Determines how metrics should be calculated based on the desired outcome e.g., a value, trend, proportion, timeframe, or compliance rate.	1 2 3 4
	Comments:	

*Graphic
Representation*

Graphical visualization of the PPI's tuples	1 2 3 4
<i>Comments:</i>	





Q4: Graphic notation of the PPICT

We will ask you to rate the clarity of the legend linked to the PPCIT model from 1 to 4. If it is not clear enough, you can modify it in the comment area.



Clarity: 1 not at all clear... 4 quite clear.

Concept	Definition	Clarity
	Optional Indicator Not Configured	1 2 3 4
	Comments:	
	Flag Required Not Configured	1 2 3 4
	Comments:	
	Indicator Optional Configured	1 2 3 4
	Comments:	

	Flag Required Configured	1 2 3 4
	<i>Comments:</i>	
	Link between the benchmark indicator and its variant	1 2 3 4
	<i>Comments:</i>	
	Link between the reference indicator and its variants allowing the intersection between the PPI's tuples	1 2 3 4
	<i>Comments:</i>	
	Link between the reference indicator and its variants not allowing the intersection between the PPI's tuples	1 2 3 4
	<i>Comments:</i>	
	Link between the reference indicator and its variants whose intersection between the PPI's tuples is complex.	1 2 3 4
	<i>Comments:</i>	

Q5: Appropriation of the PPICT

Interview: Diego takes notes and records

You have just used the PPCIT for an exercise,

- What do you think?

- Compared to your professional practices, what are the advantages?

- What was easy?

- What are the disadvantages?

- What was difficult?

- How did you use existing indicators to place the new indicators?

- What do you think of the property on tuples: "all tuples of a variant PPI before applying any aggregation or any group by are subsets smaller or equal to all tuples of its reference PPI before applying any aggregation or any group by"?

- What are the advantages of this property?

- What are the disadvantages of this property?

- What changes would you make to improve the PPICT?

Q6: Creating queries using the PPICT template.

Interview: Diego takes notes and records

You have just created queries to calculate new indicators,

- What do you think?

- What was easy?

- What was difficult?

- Compared to your professional practices, what are the advantages of this model?

- What are the disadvantages?

- Did the existing queries in the PPICT (SQL view) allow you to create the new queries?

- Which operators have you refused to create the new queries?

- How did the upper-level indicators help to add the new indicators?

- Can the PPICT (SQL view) help you understand the structure of a data model that you do not know? What for?

- From your point of view, can the PPICT (SQL view) help you build new indicators? Why?

Appendix B

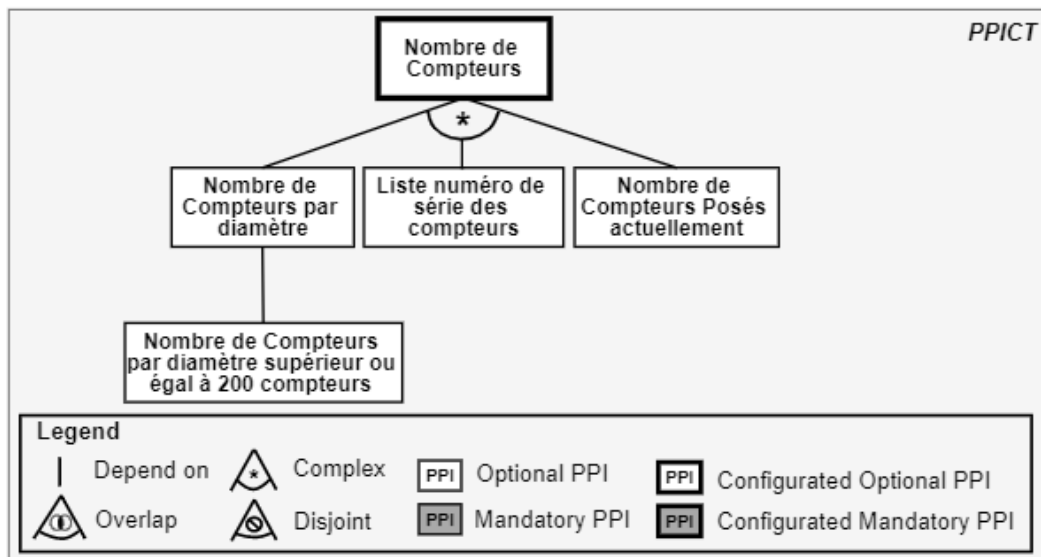
Experimental material: workbook

Description	Outils Activables et matériel expérimental
Goal	This document contains the activatable tools and experimental material to use when experimenting with the PPICT.
Diffusion	LIG et INCOM
Dated	03/06/2021
Author	Diego DIAZ
Reference Version	PPICT
Participant	

Version history			
Version	Dated	Author	Description
1.0	02/28/2020	Diego DIAZ	Document creation
1.1	04/03/2021	Diego DIAZ	Update

PPICT

1. Place the three indicators in the PPICT.
2. Modify the Symbols of the model if necessary.
3. Specify the indicator *Number of Meters by Manufacturer as mandatory*.
4. Specify the other two flags as optional.
5. Configure mandatory indicators for PPICT.
6. Configure the optional PPICT indicators associated with the diameter.

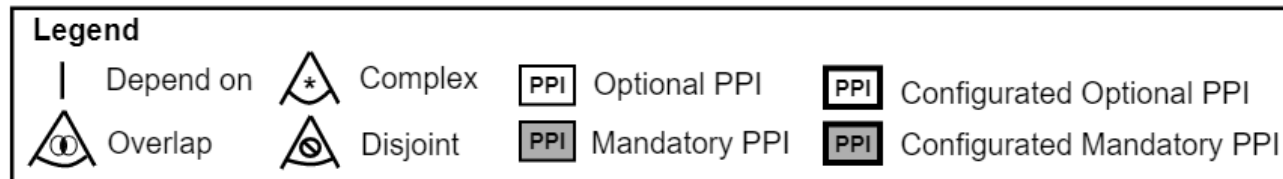
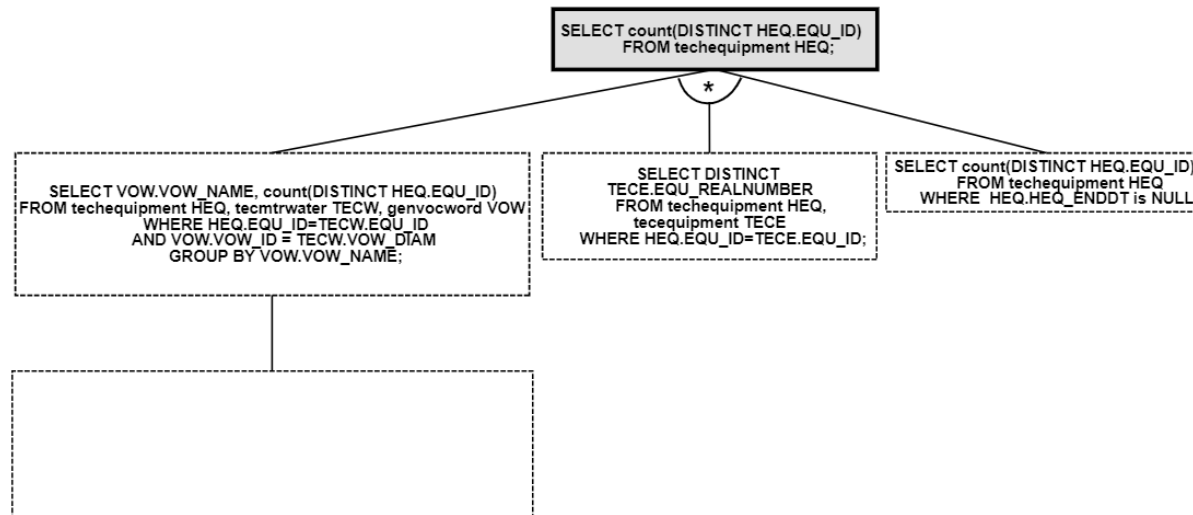


Rules for using the PPICT

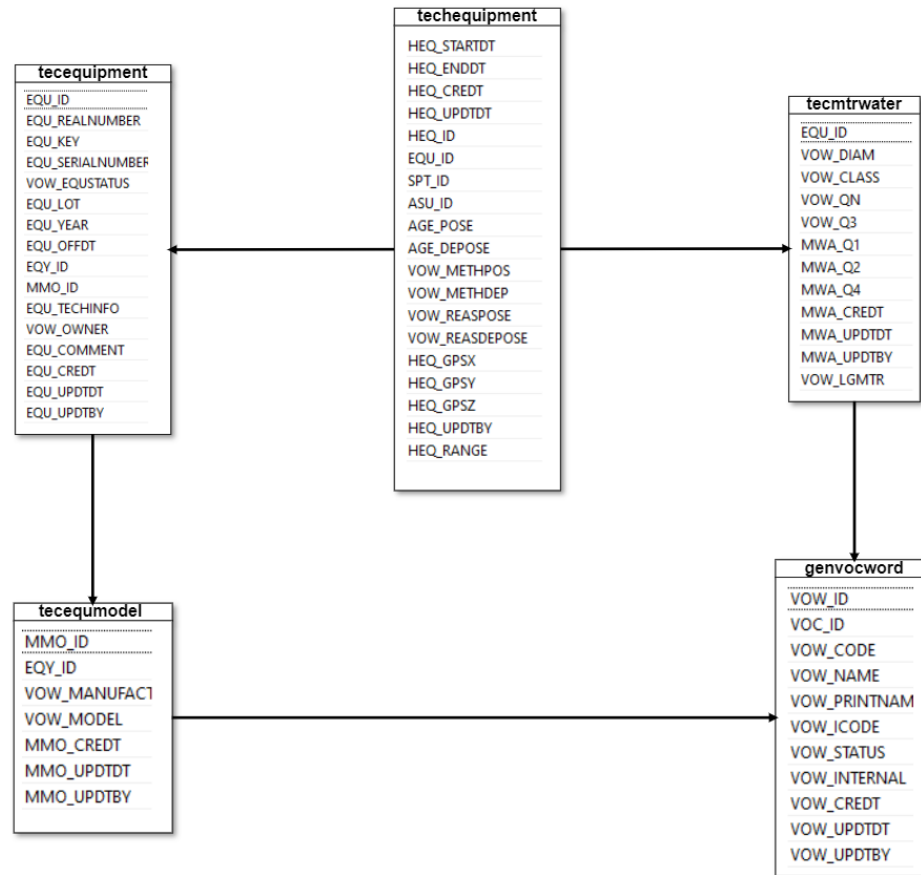
- A. All tuples of a variant PPI before applying any aggregation or any group by are subsets smaller or equal to all tuples of its reference PPI before applying any aggregation or any group by.
- B. When creating a variant indicator, it is forbidden to delete or delete operators from its reference PPI.
- C. Adding more than one variant PPI to a benchmark means that it is necessary to use the relevant symbol.
- D. The possible intersections between the variant PPIs mean that it is possible to share the tuples of these variants.
- E. The impossible intersections between variant PPIs mean that it is not possible to share the tuples of these variants.
- F. All mandatory flags must be configured, i.e., selected.

PPICT (SQL view)

Write the queries of the variant PPIs added previously using the query of the benchmark PPI.



Data model



Notice d'information et consentement

Description	Notice d'information et consentement
Goal	This document contains the information and consent notice for the experiment of the PPIC method
Diffusion	LIG et INCOM
Dated	03/06/2021
Author	Diego DIAZ
Reference Version	Mars 2020
Participant	

Version history			
Version	Dated	Author	Description
1.0	02/28/2020	Diego DIAZ	Document creation
1.0	04/03/2021	Diego DIAZ	Update

A copy of this document is given to you, another copy is kept on file.

Project: PPICT: Process Performance Indicator Calculation Tree

Researcher in charge of the project: Diego DIAZ, SIGMA - LIG - INCOM 2 Avenue de Vignate 38610 GIERES Tel.: +33 04 58 00 20 89 Tel.: +33 (0) 6 67 64 79 83 e-mail: ddiаз@incom-sa.fr

Research locations: UMR 5217 - LIG Laboratory - IMAG Building - 700 avenue Centrale - Domaine Universitaire de Saint-Martin-d'Hères Postal address: CS 40700 - 38058 Grenoble Cedex 9 - France Tel.: 04 57 42 14 00

Aim of the research project: The goal of the research project is to study supporting the variability of KPIs in families of business processes? How to calculate and model a variable KPI?

What is expected of you?

If you agree to participate in this study, we will take you to workshops to test the PPICT method. If you agree to participate in this study, you will take questionnaires and the results of your activity will be kept anonymous. An audio and video recording will be made during the experimental sessions.

Your rights to withdraw from research at any time

Your contribution to this research is voluntary. You can withdraw or cease participation at any time. Your decision to participate, to refuse to participate, or to cease participation will have no effect on your professional activity, your status and your future relations with the LIG Laboratory - IMAG Building - 700 avenue Centrale - Domaine Universitaire de Saint-Martin- d'Hères Postal address: CS 40700 - 38058 Grenoble Cedex 9 - France.

The data obtained will be treated with the utmost confidentiality; your identity will be veiled using a random anonymization number; no other information will be revealed that could reveal your identity; all data will be kept in a secure place and only the scientific manager and assistant researchers will have access to it. You have the possibility to exercise your rights with Diego DIAZ, SIGMA - LIG - INCOM 2 Avenue de Vignate 38610 GIERES Tel.: +33 04 58 00 20 89 Tel.: +33 (0) 6 67 64 79 83 e-mail: diego.diaz@incom-sa.fr

Risks

To our knowledge, this research does not involve any risk other than those of everyday life.

Diffusion

This research will be disseminated in conferences, and it will be published in conference proceedings and academic journal articles.

You can ask questions about the research at any time by contacting the project scientist by email at ddiaz@incom-sa.fr.

Consent to participate

By signing the consent form, you certify that you have read and understood the above information, that your questions have been satisfactorily answered and that you have been advised that you are free to withdraw your consent or to withdraw from this research at any time, without prejudice.

To be completed by the participant (PLEASE CHECK)

I have read and understood the above information and voluntarily agree to participate in this research.

Last name, First name - Date – Signature

Example of the experiment results

H1: the experts do not have a formalized method to calculate the indicators.

Non-formalized and non-shareable practices

- Expert 1
 - Expert 1 has five steps to calculate the indicators:
 - "The calculation of customer-oriented indicators".
 - "Determine the attributes to use (table fields)".
 - "Definition of each indicator".
 - "Application control phase using indicators".
 - "Decision-making by the client".
- Expert 2
 - Expert 2 has five steps to calculate the indicators:
 - "Frame the problem".
 - "Look at the description of the tables".
 - "Look at the data and the values of the tables".
 - "To look at the links between the tables".
 - "Calculate the indicators".
- Analysis
 - Experts do not have formal methods for calculating indicators, but they do have practices used independently based on their experiences. Expert 2 applies software development methods to the indicators.

H2: The indicators are impacted by the variability of the process and cause uncertainty on the indicators

Problem already encountered.

- Expert 1
 - "I have already encountered this problem, in particular to determine the right indicator according to the client's needs"
- Expert 2
 - "I have already encountered this problem because the indicators depend on what the customer bought, and the processes deployed".
 - "I am aware that custom developments cause code management problems".
 - "The idea is to have a standard software solution with the same parameters. However, our business involves the calculation of dedicated indicators for each client".
- Analysis
 - Experts say variability in processes leads to uncertainty over PPIs.

The indicators are impacted by the variability of the process.

- Expert 1
 - "I define the impact of variability microscopically and macroscopically. Because it is too complicated to master how a process changes from one client to another, consequently how the PPIs will have to be calculated".
- Expert 2
 - "Since it is very complicated to standardize a software solution, we have to rewrite the software components and PPIs".
- Analysis
 - Experts define the impact of Process Variability in different ways applying their knowledge of the trade. In addition, the standardization of a software product is very complex.

How the indicators are impacted by the variability of the processes

- Expert 1
 - "I know the difficulty of mastering the evolution of indicators with the variability of the process and the method of calculating PPIs because we do not know a client's work 100%".
- Expert 2
 - "The indicators are impacted by the deployed software and hardware components used by each client".
- Analysis
 - The difficulty is linked to the ignorance of a client's data and the specificity of the client's context. Furthermore, we are not sure of the software components deployed and used by our clients.

Motivation

- Expert 1
 - "I recognize that there is a difficulty and unreliability of PPIs".
- Expert 2
 - "I don't see how I could deal with these problems because software products are very complicated to standardize".
- Analysis
 - Experts agree that there is a lack of reliability in the calculation of indicators because they do not have tools for mobilizing indicators, i.e., tools allowing experts to model and calculate relationships between PPIs.

Solution created by experts.

- Expert 1
 - "The source repository is used to control the variability of the processes, the simplest is to use a standard process, when possible, in other words use basic PPIs".
- Expert 2
 - "Standardization of the code as far as possible".

- Analysis
 - Experts often use source repositories to monitor development dedicated to indicators for each client.
 - An expert believes that the easiest way to manage this problem is to standardize the software and indicators. However, when there are more and more customers, each is organized differently, standardizing the software and indicators is impossible.

H3: experts differentiate the main concepts of the PPICT.

Clarity of definitions related to the PPICT.

- Expert 1
 - "Use a more specific definition for a variable PPI, so you could split the definition of what is an unchanging indicator and what is a variable PPI".
- Expert 2
 - "A father PPI must have a son; I prefer the definition father rather than reference".
 - "I prefer the definition son instead of variant."
 - "A lot of terminology it is possible to simplify the terminology for a more intuitive understanding e.g., records. »
- Analysis
 - It is indeed possible to divide the definitions of an invariable PPI and a variable PPI.
 - An expert prefers simpler definitions like father and son as well as registrations instead of tuples.

H4: Designers take ownership of the PPCIT model and manage to add new indicators.

Advantages of the PPICT - natural language tree

- Expert 1
 - "PPI Validity ".
 - "The PPICT makes it possible to legitimize the work vis-à-vis the calculation of PPIs".

- "Master the reliability of the request because the logic of connection of the tables is better perceived".
- "The PPICT improves the quality and purpose of the request".
- "Capitalize on indicators and group them according to contexts or clients".
- "Ease the interaction between the designer and the client".
- "The PPICT has a speaking language for the customer".
- Expert 2
 - "The model is well formalized".
 - "The model has an understandable structure".
 - "It's easy to use and talking".
 - "Everything was easy on simple indicators".
 - "Adding filters is easier than adding new tables".
 - "The wording, which is to say the name, makes it possible to place an indicator very easily".
 - "The beginning of the sentence identifies the father".
 - "The inclusion rule of tuples allows users to know the information of an indicator without executing the request".
- Analysis
 - Experts have given very good comments on the tree structure in natural language, however there are limits concerning the number of PPIs per family.

Disadvantages of the PPICT - natural language tree

- Expert 1
 - "There is no vision of the request on the first tree".
 - "Problem in positioning an indicator when it has two potential fathers".
 - "Problem with the size of the tree if there are a lot of indicators".
 - "Classification of an indicator in a group that is too large no doubt difficult".
- Expert 2
 - "Adding new tables can be complicated".
 - "From second grandson the insertion of SQL code becomes easier".

- "The definition of a complex intersection needs to be clarified".
- "The terminology is not intuitive son and father".
- Analysis
 - According to expert 2: the definition of a complex intersection in the PPICT resembles the definition of a possible intersection. It is therefore essential to restructure the definition of the concept of complex intersection.

Improvement

- Expert 1
 - "Suggest possible fathers for the new indicators".
 - "Indexing of father indicators".
 - "Have a research system on the definition of indicators".
 - "Have a search system on the query for calculating indicators (table name, ID, fields to use)".
 - "Tuning helps define the structure of the request and improve performance".
 - "Indexing of tables".
 - "Identification of primary keys".
- Expert 2
 - "The inclusion rule of tuples needs to be clarified as this is more understandable as a recording".
 - "It is possible to review the definition of certain concepts such as that of father for reference, son for variant and records for tuples".
- Analysis
 - Several possible improvements were mentioned by the experts. For example, Propose the possible fathers for the new indicators thanks to a search system allowing the indexing of the tables and fields to be used, this improvement is very relevant for a PPICT having many modeled indicators. Furthermore, it is essential to review the title of the reference and variant concepts because this would be easier to interpret as father and son.

- Some improvements suggested by the experts will be implemented in future versions of the PPICT, everything that is looking for key indicators.

Reuse of indicators

- Expert
 - "Facilitates the work of the designer of indicators because it allows users to reuse queries already built based on the good definition of the inclusion rule of tuples".
 - "Allows use of the beginning of the indicator name sentence to add new indicators".
- Expert 2
 - "Reusing flags is easier when it comes to a shorter name".
- Analysis
 - The PPICT makes it easier to identify the indicators requested by customers since it is a natural language representation, designers and customers can easily identify

H5: the PPICT allows you to place the new indicators in the tree structure

PPICT tree - natural language part

- Expert 1
 - "The tree structure of the PPICT makes it easy to place the indicators because, according to the definitions given, one indicator is different from another".
- Expert 2
 - "The tree structure allows to follow a hierarchy allowing to know the data without executing the indicators".
- Analysis
 - The tree structure is intuitive, however the placement of an indicator with potentially two fathers should be clarified.

Limitation of errors

- Expert 1
 - "Having already prepared queries reduces errors by building cleaner queries".
- Expert 2
 - "The expert did not speak of limitation of errors".
- Analysis
 - Since we already have a logic of the tables, we can reduce the writing errors of the new requests.

Difficulties of the tree structure

- Expert 1
 - "You have to be able to place the indicators according to the different possible fathers".
- Expert 2
 - "When the wording is too long, we risk getting lost".
- Analysis
 - Expert 1 noted the need to be able to place the indicators more easily according to the possible fathers, especially when the PPICT is very large. Expert 2 noted a difficulty in placing the new indicators when the indicator labels are too long. One possible solution would be to suggest the possible fathers of a new indicator from the labels thanks to a search function.

Perception of graphic notation

- Expert 1
 - "it's clear."
- Expert 2
 - "Link between the indicator of reference to its variant is clear but one can add an arrow to indicate the direction".
 - "Combination of possible and impossible intersections"

- Analysis
 - In general, the experts appreciated the notation a lot because it is understandable, however certain concepts need to be improved such as that "depends on" by adding an arrow to indicate the direction of the son. Thus, it is possible to eliminate the complex intersection to leave the possible intersection.

H6: expert creates SQL queries using the PPICT.

Add requests.

- Expert 1
 - "Seeing the existing queries in the PPICT, it allowed me to use the existing SQL structures to create the new queries".
- Expert 2
 - "Take a grandson as an example and be able to place him elsewhere".
 - "Improve the definition of the complex intersection because it seems to be a possible intersection".
- Analysis
 - Regarding the PPICT-SQL tree, expert 2 mentioned that it is easier to start writing the request for a variant indicator than that for a reference indicator. That is to say, start by writing the low-level queries and then write those of the high level.
 - Regarding the PPICT-label tree, expert 1 mentioned that it is easier to start placing the new indicators from top to bottom because the labels are shorter, and therefore more compressible.
 - The PPICT is flexible and adapts to different business practices.

PPICT as a software tool

- Expert 1
 - "If the PPICT were a software tool, the expert could help place the new indicators by identifying the possible fathers".

- Expert 2
 - "I wonder how the software tool would be implemented because it would allow to use good practices for the calculation of indicators how to avoid the select between select and the links between the tables using the where".
- Analysis
 - We note that a software tool would be very appreciated for the calculation of variable indicators in the profession of software publishers.

Speed and efficiency in creating snowshoes.

- Expert 1
 - "The PPICT allows you to quickly design the SQL queries associated with the indicators, taking into account the correct syntax and definitions made by customers".
- Expert 2
 - "Regarding the indicator *Number of Meters per Manufacturer*, I took a lot of time because it is necessary to add many operators to the request. However, for the other two it was easier".
- Analysis
 - Separating the natural language part and the SQL query part makes it possible to differentiate the indicators to be viewed by a client and the design to be implemented by the developers.
 - We note that it is much easier to add certain missing operators to the indicators located at the bottom of the PPICT because in general these indicators have more information than those of the higher level.

Advantage of the PPICT SQL tree structure

- Expert 1
 - "The model becomes complex if it has a lot of indicators. Since the idea is to facilitate the work for the designers of indicators, a means should be provided to delimit the quantity of indicators per tree."

- Expert 2
 - "The simple fact of adding filters to the grandson indicators reduces the time taken to design queries while keeping good practices for calculating indicators".
- Analysis
 - Limit the number of indicators per family and can be a solution to prevent the tree structure from becoming complex.

Disadvantage of the PPICT SQL part model tree

- Expert 1
 - "The space to write requests is very small".
- Expert 2
 - "The colors linking the data model and the PPICT in SQL part, are badly put because it is possible to use one color per field".
- Analysis
 - The colors that link the data model and the PPICT SQL model are inconsistent. We could use only one field color.

H7: existing queries in the PPICT allow you to create new queries.

Existing queries optimize the design time of new queries.

- Expert 1
 - "Only the father requests allowed me to create the new requests by looking at the necessary and not wasting time in the design".
- Expert 2
 - "The grandson indicators allowed me to optimize the design time of new queries, however for the query *Number of Meters by Manufacturer*, I had to understand the data model to add the missing operators".
- Analysis
 - In conclusion, the more information we have in the PPICT SQL part model, the easier it is to add new operators. That is, designing an indicator

that is in a lower level, i.e., grandson will be easier than designing an indicator that is in a higher level, i.e., father.

- The design time for a higher-level indicator is higher than the design time for a lower-level indicator.

Language PPICT party SQL speaking

- Expert 1
 - "This removes the fact of looking elsewhere because it is a speaking language, and we quickly find our way in the model to be able to define the fields to use".
- Expert 2
 - "It's still SQL language. For people who master SQL. It is relatively easy. However, it is necessary to check how we could reuse the operators missing from an indicator of the upper level but present in an indicator of the lower level".
- Analysis
 - The syntax of the PPICT SQL model did not have a problem of comprehension. However, it is necessary to check how to help the experts to more easily add the operators of a child indicator of the higher levels.

H8: Understanding of the data model as a result of using the PPICT.

Level of understanding of the data model

- Expert 1
 - "The model allows us to remember the structure of the database and contextualizes us on the domain. The model offers ready-made design bricks to save time in building queries and understanding the logic of tables".
- Expert 2
 - "This model allowed me to learn how to calculate the indicators associated with counters in the Anemone models".
 - "More indicators exist in the PPICT more information we will have to understand the data model".

- Analysis
 - Even if some indicators were more complicated to calculate than others, we find that the experts managed to understand the model concerning the fields used in each indicator.

Validation and automatic inclusion of indicators

- Expert 1
 - "It is a tool for modeling and designing indicators for beginners and data model experts".
- Expert 2
 - “Automatic inclusion is based only on the father's SQL structures. This represents a reduction in the time spent on the design and inclusion of existing operators”.
- Analysis
 - In conclusion, the PPICT is a tool that is used to build indicators that can be used by beginners or SQL experts according to the opinion of an expert.
 - Automatic inclusion of the father's SQL structure is essential.
 - Possible improvements to the PPICT.
 - Definition.
 - The notion of Tuples needs to be clarified as this is more understandable as records.
 - Review the definition of Reference and Variant PPI because it is more understandable to speak of Father and Son.
 - PPICT-label.
 - Suggest possible references for new indicators to add.
 - Research system on the definition of PPI.
 - PPICT-SQL.
 - Search system on existing queries by table name and fields to identify potential operators to use.
 - Define the request structure to improve performance.
 - Indexing of existing query tables.
 - Identification of primary keys of existing requests.