Enhancing the Modelling Perspective of Process Performance Management

Irene Bedilia Estrada Torres

PhD Dissertation

Supervised by: Dr. Manuel Resinas Arias de Reyna
and Dra. Adela del Río Ortega

Universidad de Sevilla

September 2018

Support:

Pre-doctoral scholarship has been granted by Andalusian Local Government under project COPAS (P12-TIC-1867). Additional support for research visits provided by the Marie Sklodowska-Curie grant agreement No. 645751 (RISE_BPM); and for attending conferences provided by: the European Commission (FEDER); Spanish and Andalusian R&D&I programmes (grants TIN2012-32273 (TAPAS), TIC-5906 (THEOS), P12-TIC-1867 (COPAS), TIN2015-70560-R (BELI)).

Dr. Manuel Resinas Arias de Reyna, Profesor Contratado Doctor del Área de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla y Dña. Adela del Río Ortega, Profesor Ayudante Doctor del Área de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla

HACEN CONSTAR

que Dña. Irene Bedilia Estrada Torres, Máster Universitario en Ingeniería y Tecnología del Software, ha realizado bajo nuestra supervisión el trabajo de investigación correspondiente a su Tesis Doctoral titulada:

Enhancing the Modelling Perspective of Process Performance Management

Una vez revisado, autorizamos el comienzo de los trámites para su presentación como Tesis Doctoral al tribunal que ha de juzgarlo.

Fdo: Dr. Manuel Resinas Arias de Reyna y Dra. Adela del Río Ortega
Sevilla, septiembre de 2018

Yo, Dña. Irene Bedilia Estrada Torres con NIE número Y2523772L,

DECLARO

mi autoría del trabajo que se presenta en la memoria de esta Tesis Doctoral que tiene por título:

Enhancing the Modelling Perspective of Process Performance Management

Lo cual firmo en Sevilla, en septiembre de 2018.

Fdo: Dña. Irene Bedilia Estrada Torres

To Pablo.

You motivated me to start this.

Thank you for that and for everything you helped me get.

Acknowledgments

During all this time many people have been involved in my doctoral process and although I will make an effort to mention them all, I anticipate my apologies if I expressly fail to mention any of them.

First of all, thank God for giving me the opportunity to live this new experience and possibilities of carrying out this thesis.

Second, I want to thank my tutor and supervisors, Antonio Ruiz, Adela del Río and Manuel Resinas. Thank Antonio, because you were the first one to trust me and open the doors of the research group so that I could be part of it. Thank Manuel and Adela, because this thesis is the result of their advice, accompaniment, guidance and support. Thank you for introducing me to the research world, for your patience and for your company at all stages of this work since its inception.

To the members of the ISA group, thank you for your kindness and for making me feel part of the research group.

I would also like to express my sincere thanks to the Department of Applied Informatics of the Federal University of Rio de Janeiro. Especially thanks to Flavia Santoro and Fernanda Baião for sharing their ideas with me and for giving me the opportunity to work with them during my research stays in Rio de Janeiro, Brazil.

Thanks to José Juan Romero and the Jaime Loring Foundation (Córdoba) for trusting me and supporting me in starting this thesis process.

Finally, I would like to say a few words of thanks in Spanish. Quiero agradecer especialmente a familia, mis padres, Marta y Ernesto, y a mi hermano Ernesto, porque desde la distancia siempre han confiado en mi, me han dado su apoyado incondicional y se han preocupado por acompañarme y animarme en todo. A Maricarmen, porque desde hace varios años se ha convertido en mi hada madrina y su cariño y apoyo han estado siempre presentes. A mi familia adoptiva en España, Báez González, por su confianza y cariño que me han hecho más fácil el andar en este proceso. Por último, gracias a ti, Pablo, por haber sembrado en mí la semilla de la curiosidad y porque desde el principio me has ayudado, apoyado e impulsado a seguir; porque siempre has confiado y creído en mí, incluso más que yo; por tu paciencia y por estar siempre a mi lado. Te debo mucho, y esto también te lo debo a ti.

I.B.E.T.

Abstract

Process Performance Management, which comprises the stages of planning, monitoring and controlling of business process performance, is considered an essential part of business process management and provides a detailed understanding of how business processes can be designed and redesigned to improve their performance. The performance of business processes is usually measured by means of process performance indicators (PPIs), that are evaluated and monitored with the aim of identifying whether strategic and operational goals are being achieved. The same as business processes, and actually together with them, these PPIs must be defined and modelled, so in this dissertation, we focus on their *Modelling Perspective*. This perspective allows us to describe in detail all PPIs associated with a business process, specifying the set of attributes that define them and the information that needs to be obtained from the business process for their computation.

Most proposals related to the performance measurement of business processes have focused on measuring the performance of structured, repetitive and highly defined business processes. Changes and new requirements of businesses have given rise to new needs in the management of business processes, requiring more flexibility for instance, to manage collections of business process alternatives and knowledge-intensive and highly dynamic processes. However, current techniques of process performance management have not evolved at the same pace as business processes. Existing proposals cannot be used "as-is" in scenarios with business processes that require more flexibility because those processes have different nature. For example, those proposals are not able to define PPIs on collections of business processes or to define PPIs taking into account process elements that are not usually present in traditional processes, such as decisions, interactions or collaborations between participants. This generates a gap between business processes and the measurement of their performance.

In order to face this challenge, this dissertation seeks to extend the current boundaries of process performance measurement. To do so, we based on existing and well-founded techniques, mainly applicable to structured business processes whose behaviour is mostly known a priory, and extend them to cope with the new identified requirements. Specifically, we propose a set of artefacts that focus on the definition and modelling perspective of performance measures and indicators.

First, we propose the extension of the PPINOT metamodel, a metamodel previously used for defining and modelling of process performance indicators, in four directions: the total or partial reuse of PPI definitions; the modelling of PPIs taking into account the variability of business processes and PPIs themselves; the definition of PPIs on knowledge-intensive processes and the relationship of the definition of PPIs to a particular set of business process elements, such as decisions. The extensions of the metamodel are designed to work with relevant proposals for each one of the areas addressed, such as the Knowledge-Intensive Process Ontology (*KIPO*), an ontology that provides constructs for defining Knowledge-intensive Processes; the Decision Model and Notation (*DMN*), a standard for defining and modelling of decisions; *Provop* and *C-iEPC*, two business process modelling languages associated with variability. To facilitate the representation of PPI definitions and models, we also propose the extension of the two PPINOT notations: Visual PPINOT, a graphical notation and the template based notation that uses linguistic patterns for PPI definitions. In addition to those extensions, we provide two more contributions: a methodology based on the integration of PPINOT and KIPO, and on concepts of lead and lag indicators with the aim of guiding participants in the implementation of PPIs in accordance to business goals, and guidelines in the form of a set of steps that can be used to identify decisions that affect the process performance. Finally, we have also taken the first steps to implement the metamodel extensions to the PPINOT modelling tools. Specifically, we modified the graphical editor to facilitate the modelling of PPIs by reusing PPI definitions.

We validated the proposals presented through different scenarios and case studies: by analysing the processes and performance measures of the Supply Chain Operations Reference (SCOR) model, using measures and indicators provided by the Andalusian Health Service (Spain) and by means of a case study in an information and communication technology outsourcing company in Brazil.

Our proposal of "Enhancing the Modelling Perspective of Process Performance Management" allows us to: reuse total or partial PPI definitions in different PPIs or business processes; define PPIs in a collection of business processes (process family) taking into account the variability of processes and PPIs themselves; define PPIs over Knowledge-intensive processes; use a methodology to guide participants in the implementation of PPIs in accordance to business goals; analyse the impact of decisions related to business processes on PPIs; define decision performance indicators (DPIs) to measure the performance of decisions related to business processes; and use process performance information in the definition of decisions.

Resumen

La *Gestión del Rendimiento de los Procesos*, que comprende las etapas de planificación, supervisión y control del rendimiento de los procesos de negocio, se considera una parte esencial de la gestión de los procesos de negocio y proporciona detalles de cómo se pueden diseñar y rediseñar los procesos de negocio para mejorar su rendimiento. El rendimiento de los procesos de negocio suele medirse por medio de indicadores de rendimiento de procesos (PPI, por sus siglas en inglés), que se evalúan y monitorizan con el fin de determinar si los objetivos estratégicos y operativos están siendo alcanzados. Al igual que los procesos de negocio, y en realidad junto con ellos, estos PPIs deben ser definidos y modelados, por lo que en esta tesis, nos centramos en su *Perspectiva de Modelado*. Esta perspectiva nos permite describir en detalle todos los PPIs asociados a un proceso de negocio, especificando el conjunto de atributos que los definen y la información que se necesita obtener del proceso de negocio para su cálculo.

La mayoría de las propuestas relacionadas con la medición del rendimiento de los procesos de negocio se han centrado en la medición del rendimiento de los procesos de negocio estructurados, repetitivos y altamente definidos. Los cambios y los nuevos requisitos de las empresas han dado lugar a nuevas necesidades en la gestión de los procesos de negocio, que requieren más flexibilidad, por ejemplo, para gestionar colecciones de alternativas de procesos de negocio y procesos altamente dinámicos e intensivos en conocimientos. Sin embargo, las técnicas actuales de gestión del rendimiento de los procesos no han evolucionado al mismo ritmo que los procesos de negocio. Esas propuestas no pueden ser utilizadas "tal cual" en escenarios con procesos de negocio que requieren más flexibilidad porque esos procesos tienen una naturaleza diferente. Por ejemplo, esas propuestas no son capaces de definir PPIs en colecciones de procesos de negocio o de definir PPIs teniendo en cuenta elementos del proceso que no suelen estar presentes en procesos tradicionales, tales como decisiones, interacciones o colaboraciones entre participantes; generando así una brecha entre los procesos de negocio y la medición de su rendimiento.

Para hacer frente a este reto, esta tesis pretende ampliar los límites actuales de la medición del rendimiento de los procesos. Para ello, nos basamos en técnicas ya existentes y bien fundamentadas, principalmente aplicables a procesos de negocio estructurados cuyo comportamiento se conoce en su mayor parte a priori, y los ampliamos para hacer frente a los nuevos requisitos identificados. Específicamente, proponemos un conjunto de artefactos que se centran en la definición y la perspectiva de modelado de las medidas e indicadores de rendimiento.

Primero, proponemos la extensión del metamodelo PPINOT, un metamodelo previamente utilizado para la definición y modelado de indicadores de rendimiento de procesos, en cuatro direcciones: la reutilización total o parcial de definiciones de PPIs; el modelado de PPIs teniendo en cuenta la variabilidad de los procesos y de los propios PPIs; la definición de PPIs sobre procesos intensivos del conocimiento y la relación de la definición de PPIs con un conjunto particular de elementos de los proceso, como las decisiones. Las extensiones del metamodelo son diseñadas para trabajar con propuestas relevantes en cada una de las áreas abordadas, como KIPO (Knowledge-Intensive Process Ontology), una ontología que proporciona constructores para la definición de procesos intensivos en conocimiento; DMN (Decision Model and Notation), un estándar para la definición y modelado de decisiones; Provop y C-iEPC, dos lenguajes de modelado de procesos de negocio asociados a la variabilidad. Para facilitar la representación de las definiciones y modelos de PPIs, también se propone la extensión de las dos notaciones del PPINOT: Visual PPINOT, una notación gráfica y la notación basada en plantillas que utiliza patrones lingüísticos para las definiciones de PPI. Además de estas extensiones, también proponemos dos contribuciones más: una metodología basada en la integración de PPINOT y KIPO, y en los conceptos de indicadores *lead* y *lag* con el objetivo de guiar a los participantes en la implementación de PPIs de acuerdo a las metas de negocio, y directrices en forma de un conjunto de pasos que pueden utilizarse para identificar las decisiones que afectan el rendimiento del proceso. Finalmente, también hemos dado los primeros pasos para implementar las extensiones del metamodelo en las herramientas de modelado de PPINOT. Específicamente hemos modificado el editor gráfico para facilitar el modelo de PPIs por medio de la reutilización de definiciones.

Hemos validado las propuestas presentadas a través de diferentes escenarios y casos de estudio: analizando los procesos y las medidas de rendimiento del modelo de referencia SCOR (Supply Chain Operations Reference), utilizando medidas e indicadores proporcionados por el Servicio Andaluz de Salud (España) y a través de un caso de estudio en una empresa de outsourcing de tecnologías de la información y la comunicación en Brasil.

Nuestra propuesta de “Mejorar la Perspectiva de Modelado de la Gestión del Rendimiento de Procesos” nos permite: reutilizar definiciones totales o parciales de PPIs en diferentes PPIs o procesos de negocio; definir PPIs en una colección de procesos de negocio (familia de procesos) teniendo en cuenta la variabilidad de los procesos y de los propios PPIs; definir PPIs sobre procesos intensivos en conocimiento; utilizar una metodología para guiar a los participantes en la implementación de PPIs de acuerdo con los objetivos de negocio; analizar el impacto sobre los PPI, de las decisiones relacionadas con los procesos de negocio; definir indicadores de rendimiento de las decisiones (DPIs) para medir el rendimiento de las decisiones relacionadas con los procesos de negocio; y utilizar la información sobre el rendimiento del proceso en la definición de decisiones.

Contents

LIST OF FIGURES

List of Tables

Part I

PREFACE

Introduction

"The first step, my son, which one makes in the world, is the one on which depends the rest of our days."

Voltaire (1694-1778),
French writer, deist and philosopher

In this dissertation, we report on our contributions to develop a set of techniques to enhance the process performance management of the modelling perspective of business processes by means of the extension of PPI definitions and their application to different types of processes and process perspectives. In this chapter, we first introduce the research context and the purpose of this research in Section §1.1. Section §1.2 explains the problem addressed and establishes the goals defined for this proposal. The summary of our contributions resulting from the approaches followed to the consecution of the goals are presented in Section §1.3. Section §1.4 describes the research method followed. Section §1.5 explains the context in which the work of this dissertation has been performed. Finally, in Section §1.6, we present the structure of this dissertation.

1.1 Research Context

Organisations define business processes with the aim of contributing to the achievement of their organisational goals. In order to determine whether those established goals are being effectively achieved, it is necessary to monitor the performance of those business processes [165]. If, as a result of this monitoring, the performance information obtained is very different from the results expected and defined through the performance goals, it is necessary to analyse the business process to identify possible improvement areas that, after adjustments and configurations, contribute to obtaining better performance results [162].

Most approaches focused on measuring the performance of business processes are based on the identification, definition, computation and evaluation of process performance indicators (PPIs). PPIs are quantifiable metrics that allow the evaluation of efficiency and effectiveness of business processes taking and using data that is generated within the process flow [32]. Several performance dimensions such as time, cost and quality, can be refined by means of PPI definitions, assuming that the data to calculate PPIs is available [41]. Over time, different techniques have been developed with the aim of managing process performance, such as [31, 130, 151]. However, although business needs have led to an evolution of business process management, much of the evolution has focused on the control flow perspective, where many alternatives have emerged to define, model and manage new business processes and very few proposals have focused on process performance management.

Organisations typically define a large number of business processes [36]. Several authors agree that the modelling of a large number of business processes can become a complex and error prone task [50, 99, 169]. Reuse is proposed as a technique to reduce the time spent on creating a solution from scratch, by reducing efforts, time and costs; and it also reduces the possibility of introducing errors during their management. In business process modelling, reuse is a technique that allows us to design business processes by using existing process models [99]. The reuse of business processes has a beneficial impact on the quality, efficiency and effectiveness of business process modelling as it seeks to reduce modelling time by avoiding modelling the same business process or part of it multiple times [77, 99].

Abstraction is one of the most used techniques to materialise the reuse and it is considered an essential feature in any reuse technique [86]. In business process modelling, abstraction helps reduce complexity in models that must be presented to others, facilitates readable processes, and allows the representation of business processes by means of high level views, showing aggregated activities and hiding irrelevant details to particular users [39, 148, 149]. The problems arising from the management of the control flow of business processes are also identified in the performance perspective, such as the redundancy in the definition of indicators and multiple drawbacks derived from it. However, the techniques used to address these problems can be analysed from the point of view of performance.

In addition to the redundancy of information in business process models, changes and new requirements of businesses have given rise to new needs in the management of business processes, requiring more flexibility. Four flexibility needs, *variability, looseness, adaptation, and evolution* are proposed as the main categories of a taxonomy of process flexibility needs that may affect each perspective of a business process (i.e., behaviour, organisation, information, operation, function, time) [137]. According to this proposal, each need identified can be associated with technical requirements that a Process Aware Information System (PAIS) should facilitate for the management of processes related to each category. In this research, we focus on the first two categories of the taxonomy: variability and looseness as we consider them to be the two most related needs to the characteristics of the business process, while the other two, adaptation and evolution, are more dependent on the characteristics of the PAIS.

Process *Variability* is related to a set of processes with similarities between them (process variants), which usually share the same core process. A PAIS should be able to support configuration of process models and process variants. In this context, a *Customisable process* is defined as a business process that "represents a family of process variants in a way that a model of each variant can be derived by adding or deleting fragments according to customization options or according to a domain model" [90]. These types of processes can be used in two ways. First, to gather all possible process variants into a single model and then eliminate those parts that do not belong to a particular configuration (variability by restriction). Or second, to build a base configurable model with only the information common to all or most of the variants and that later will have to be configured by adding the required additional information (variability by extension). *Looseness* characterises loosely specified processes. A loosely specified process is defined as "a process model which is not fully prespecified, but keeps some parts unspecified at build-time by deferring decisions to run-time" [137]. Loosely specified processes are considered a type of Knowledge-intensive Processes (KiPs), which are characterised by being non-repeatable, unpredictable and emergent processes and also because only their goals are known a priori. KiPs are processes that require flexibility at design and at run time, and whose behaviour "is heavily dependent on knowledge workers performing various interconnected knowledge-intensive decision making tasks" [35].

Since Knowledge-intensive Processes are based on participants' knowledge, different authors stress the importance of the decisions made in the process to the point of defining concrete concepts to represent these decisions in their models or to take into account the information related to them [137, 145]. As a result of the evolution of the business and the new requirements that must be addressed by the business processes, not only has the way of managing or modelling the process changed, but also some of its components or perspectives have evolved. Decisions made in a business process, which have traditionally been defined and modelled either as part of the business process or through a set of business rules, are an example of this. Given their importance and the evolution of the business processes themselves, recently standards and notations have been released with the aim of providing constructs to model decisions and decoupling decisions from process models. These decisions, in turn, can also be related to the characteristics of variability and looseness previously described.

Most proposals related to the performance measurement of business processes have focused on measuring the performance of structured, repetitive and highly defined business processes, whose logic is known before their execution and which cannot be used “as-is” in other scenarios that require more flexibility. This situation limits the application of those proposals in real-world cases or scenarios, as in many cases business processes are much more complex and must be able to adapt to the particular situations of each organisation.

The analysis of performance measurement on customisable processes may help to identify how variations in the business process control flow influences the definition of PPIs. It includes determining, among other things, what business process elements can provide information for the computation of PPI values, how these PPI values can actually be computed or what can be points of variability in PPI definitions. Furthermore, despite the growing body of research focused on understanding KiPs, and on proposing systems to support them, only preliminary works exist when it turns to evaluate performance of KiPs (e.g., [141]). There is a lack of a generic proposal that allows the measurement of different concepts related to knowledge, identifies their influence and synergy upon traditional measures and, in turn, evaluates the overall performance of KiPs. Performance information provided by PPIs may be complemented or be influenced by means of other performance indicators defined over different business process elements, such as *decisions*; and the relationship between PPIs and other performance indicators can be a topic for discussion that has not been addressed until now. Hitherto, certain proposals have been done to analyse performance from the process perspective of data [12] or resources [113], but very little is known about the relationship between performance and the decisions made in the business process.

1.2 Thesis goals

This dissertation seeks to extend the current boundaries in the application of the performance measurement of business processes, to focus not only on the enhancement of current alternatives for measuring structured business processes whose behaviour is mostly known a priori, but also on proposing alternatives for the modelling of process performance considering flexibility in business processes.

The main goal of this dissertation is to provide techniques and mechanisms to improve the current performance management of business processes from a modelling perspective by means of the enhancement of PPI definitions and their management considering flexibility in business processes.

This challenge can be stated by the following four research questions:

Research question RQ-1. How can the modelling perspective of process performance management be improved taking into account the reuse in PPI definitions?

Research question RQ-2. How does the variability of business processes affect the modelling of PPIs?

Research question RQ-3. How can PPIs be modelled in KiPs?

Research question RQ-4. Which are the relationships between decisions and performance of business processes?

With each of the four questions posed, we intend to address four different approaches identified and related to the modelling perspective of the performance management of business processes.

With regard to *research question RQ-1*, process performance indicators are usually modelled one by one for each business process where they are defined, regardless of whether those indicators are totally or partially used by other processes. The aim of this question is to identify and propose an alternative that allows the definition of indicators by reusing all or part of their definitions and to identify how, based on the current PPI definitions, performance measurement can be improved using reuse techniques, which to date have only been applied to the definition of the process model.

Changes and variations in business processes can be reflected in different process perspectives, such as control-flow, data or resources [88]. However, as far as we know, this variability has not been addressed from a performance perspective and very little information relates the measurement of process performance to variability. In this sense, through *research question RQ-2* we want to analyse how variability affects the performance perspective of business processes from the definition of PPIs, as well as to propose alternatives to model PPIs related to a set of process variants.

Research question RQ-3 seeks to analyse the characteristics of KiPs, which are different from structured processes, in order to identify how the elements of this type of processes can be measured, what steps should be taken into account for their definitions, as well as to propose mechanisms for modelling PPIs in KiPs.

In KiPs the importance of the participants' knowledge is evident and the decisions they make are decisive in the execution of the process. Decisions are considered a particular set of business process elements, which, given the evolution of business process models, have become an element of interest for analysing process performance. The purpose of *research question RQ-4* is to analyse the impact of decisions on process performance, the performance measurement of decisions themselves and how process performance measures and indicators can be used for the definition of decisions.

1.3 Summary of Contributions

This research is focused on the development of artefacts that improve the process performance management by means of the modelling perspective. Several scenarios were analysed to determine how PPIs can be defined extending current boundaries related to their definitions and range of application.

We summarise the contributions of this thesis, by means of two diagrams. The first, shown in Figure §1.1, indicates that contributions of this thesis are focused on the performance perspective of business processes. The proposal is divided on four blocks, one for each area addressed: reuse, variability, KiPs and Decisions. Within each block we indicate the specific element of PPINOT that has been extended, which is marked with the symbol of two crossed bi-directional arrows. The single bidirectional arrows indicate that a PPINOT element has been integrated with other proposals or has given rise to a new artefact. For example, in the variability block, both the PPINOT and metamodel notations have been integrated with two business process modelling languages, and in the KiPs block, the integration of the PPINOT and KIPO metamodel have given rise to a new ontology and a new methodology.

Figure §1.2 shows information similar to that presented in Figure §1.1, but in this case we emphasise contributions presented and their relationship with other proposals. In this figure, the central element is PPINOT, which extends in four directions. The upper and left blocks are related to variability and reuse, while the right and lower blocks are related to KiPs and decisions, respectively. In each block we highlight the extensions and/or new proposals provided. For example, in the block of reuse we extend PPINOT, while in the KiPs block, we highlight the two main contributions and their relationship with KIPO. Below we describe in detail the contributions of this thesis, classifying these contributions according to the extended or generated artefacts.

Metamodel. The proposed metamodel is not started from scratch. The *PPINOT Metamodel* (*PPINOT*) [31], which is used to model PPIs over business processes, was extended with measures and measurable elements to improve the definitions and applicability of PPIs to different scenarios and also to include new types of performance indicators. The metamodel is extended:

i. To include reuse and abstraction concepts to facilitate the reuse of PPI definitions (or parts of them) in multiple business processes and to reduce modelling complexity in PPI definitions.

ii. To represent how a PPI varies depending on the business process variant where it is defined and depending on attributes required to define it.

iii. To integrate the metamodel with a Knowledge-intensive Process ontology (KIPO) to measure the performance of KiPs; and also to introduce concepts to define PPIs in CMMN cases.

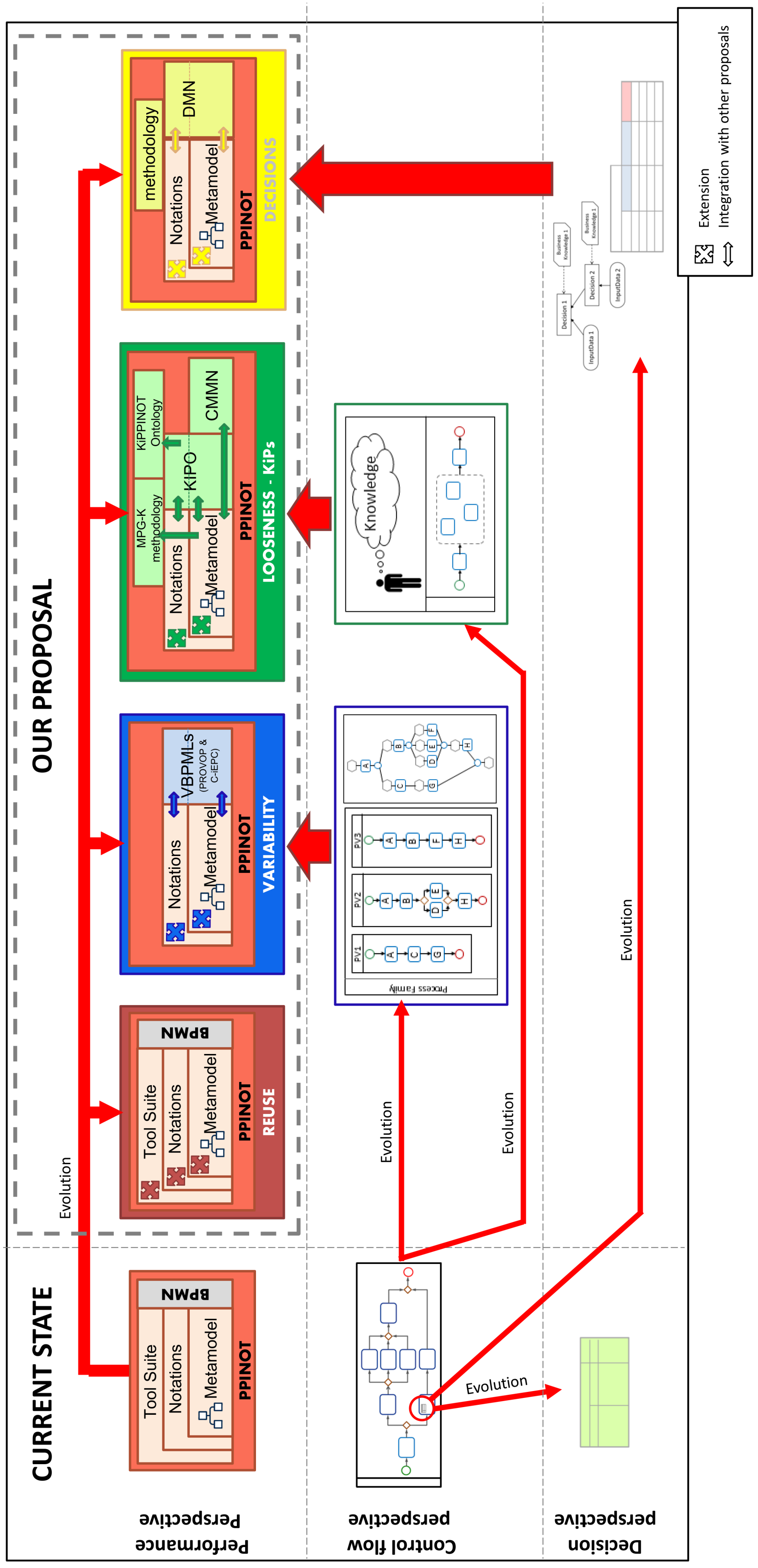


Figure 1.1: Summary of the contributions of this thesis and the area where they are applied.

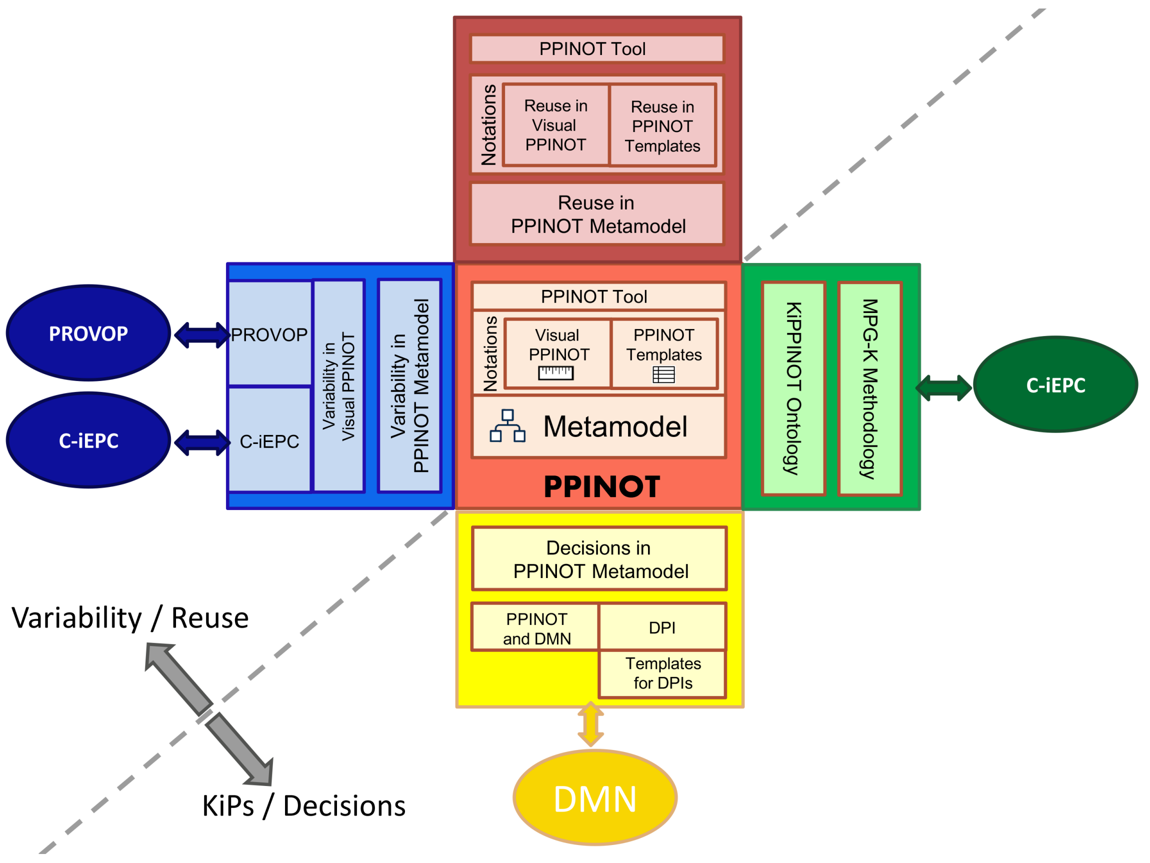


Figure 1.2: Summary of the contributions of this thesis and their relationship with other proposals.

iv. To consider concepts related to decisions as a new element from which performance information can be taken. With this extension we identify and analyse the relationship and impact of decisions on the process performance, the performance measurement of decisions themselves and the use of PPIs as input information to define decisions.

In this proposal, in addition to the graphic representation of the metamodel by means of UML diagrams, we propose a *formal definition* of PPINOT to represent the components and constraints related to the original metamodel and the extensions proposed in this thesis; and also for modelling PPIs.

Notations. PPIs can be represented using different mechanisms. On the basis of the PPINOT Metamodel, there are two notations: *Visual PPINOT*, as graphical notation, and the *template based notation* that uses *linguistic patters* for PPI definitions. In this proposal, we expanded the two PPINOT notations to adapt the modelling of PPIs to the process characteristics or specific performance measurement situations to be addressed, as discussed in the research questions. In other cases, other existing notations can also be extended to integrate PPI definitions in their context of application. Specifically, contributions of notations are detailed below.

i. The two alternatives, the graphical notation and the template based notation, are used to model PPIs taking into account abstraction concepts to facilitate the reuse of PPI definitions or parts of them.

ii. The graphical notation is extended to model variability related to PPI definitions. In this extension, the graphical notation is extended in two ways, to be adapted to two different graphical notations well-known in the context of modelling variability in business processes.

iii. Related to KiPs, we do not focus on the extension of PPINOT notation, but we modelled PPIs using the UML proposal used to model Knowledge-intensive Processes because KIPO does not have a graphical notation.

iv. Related to decisions the template based notation of PPINOT was extended to be able to define PPIs taking into account information related to decisions. In addition, the DMN standard, specifically the definition of its decision tables, was also extended to incorporate in its notation the performance information.

Methodology. A methodology based on the integration of PPINOT and KIPO (the Knowledge-intensive Process Ontology), and on concepts of lead and lag indicators, was proposed to guide participants in the implementation of PPIs in accordance to business goals. With respect to the relationships between decisions and performance, we identified a set of steps that can be used to identify decisions that affect process performance and that are manifested through PPIs that do not reach their target values.

Tools. PPINOT is a set of tools and techniques for the definition and automated analysis of process performance indicators. We extended two of those tools in relation to one of the issues addressed in this research. First, the graphical editor of PPINOT was extended to include abstraction concepts and be able to reuse PPI definitions. The second extension was applied to core application of PPINOT. Java projects was modified as a first step for the automatic analysis of PPI models using abstraction concepts, implementing new measuring elements and applying validation in accordance with the PPINOT metamodel.

Evaluation. For the evaluation of the proposals, different scenarios were used based on: the SCOR reference model, which provides a set of processes and performance measures; information of PPIs related to the IT incident management process of the Andalusian Health Service in Spain; and the case study based on an IT incident management process of a Brazilian company.

1.4 Research Methodology

In this dissertation, the *Design Science Research Methodology* (DSRM) [126] was followed. DSRM defines six activities, which are described in the context of this research next.

1. *Problem identification and motivation* is addressed by means of both a literature review used to analyse the state of the art of process performance measurement, and the study of several cases from which difficulties and drawbacks about the performance management as well as business process perspectives that can be object of interest, were identified.

2. *The objective for a solution* is to provide techniques and tools to improve the current performance management of business processes from a modelling perspective by means of the improvement of PPI definitions and their management.

3. The *design and development* of artefacts include: a metamodel that implements features to improve PPI definitions and facilitates its application in different type of business processes and scenarios; extensions of existing notations (graphical notation, templates, etc.) for modelling PPIs taking into account characteristics of the different scenarios and types of processes addressed; methodologies to provide guidelines that conduct the implementation of business processes and PPIs in accordance with organisational performance goals; and extensions of a software tool to model PPIs according to one of the approaches presented in this document.

4. The *Demonstration* phase involves the implementation of the artefacts described in the previous item in the development of software tools, prototypes or formal definitions to facilitate the PPI modelling and management, taking into account their definitions and applicability in different business processes and process perspectives.

5. The *Evaluation* phase is carried out by means of the use of PPI definitions in real scenarios to compare and analyse their feasibility and contributions. Some of those scenarios are based on: (i) the SCOR model [6] that is a reference model that provides guidelines for the supply chain management in enterprises. It was used as scenario related to reuse of PPI definitions and to manage variability in business processes; (ii) processes and indicators related to the incident management process of the Andalusian Health Service. This information was used in our analysis related to variability and decisions; and (iii) a case study based on the incident troubleshooting process within an Information and Communication Technology Outsourcing Company of a real organisation in Brazil.

6. Finally, in the *Communication* phase, the main target was the participation in national and international conferences, as well as publications in scientific journals related to business processes management and process performance.

1.5 Thesis Context

This thesis has been developed in the context of the *Business Process Management* research area of the *Applied Software Engineering (Ingeniería de Software Aplicada - ISA)* research group of the University of Seville (Spain). As a holder of a pre-doctoral research scholarship the work related to this thesis has been developed in the scope the of following research projects:

- *COPAS: ECosystems for Optimized Process As a Service.* Excellence Project of the Andalusian Local Government, referenced as P12-TIC-1867. In the context of this project I was awarded a three-years grant for the development of my PhD thesis. Specifically, my participation in this project has been related to the concrete objective 1 - *Enable the development of information systems based on PRaaS (Process as a Service)*, which consists of developing a set of techniques and methodologies that support the development of information systems based on PRaaS as well as a set of operations that allow for a holistic analysis to support decision making.

- *RISE_BPM: Propelling Business Process Management by Research and Innovation Staff Exchange.* Project founded by the European Commission from 01-05-2015 to 30-04-2019, referenced as H2020-645751. *"The RISE_BPM project is aimed at networking world-leading research institutions and corporate innovators to develop new horizons for Business Process Management (BPM)"*[112]. Thanks to our participation in this project, it was possible to obtain funding for two research stays in Rio de Janeiro, Brazil, with a total duration of 3 months. From the collaborative works of these stays, results related to the performance measurement of Knowledge-intensive Processes were obtained. My participation in this project was related to two work packages (*wp*), one in each phase of the project. The first one, the *WP3 - Technological Enablers - Real-Time Computing* and the second one, *WP9 - Synthesis of IT Artefacts - BPM Analysis*.

- *BELI: Tecnologías para Servicios Cloud Híbridos, Altamente Configurables y Regulados por Ans.* Project founded by the Ministry of Ministerio de Economía y Competitividad, referenced as TIN2015-70560-R. My participation in this project was related to two specific objectives of the project *(OBJ)*. The first one, the *OBJ1 - Define and analyse models to design H2CS*, which is related to the definition of a variability model that allows the modelling of highly configurable processes. The second one, *OBJ4 - Develop techniques to automate or guide decisions during service operation to optimise costs.* This is related to analysis made in the context of Knowldge-intensive Processes and the methodology base on lead and lag indicators and to the analysis of relationships between decisions and performance.

1.6 Structure of this dissertation

This dissertation is organised as follows:

Part I: PREFACE. It comprises this introduction chapter, in which we describe the motivation and context of our research, introduce the problems addressed and the goals pursued, describe the research methodology followed and summarise our contributions.

Part II: BACKGROUND INFORMATION. This part provides the reader with background information essential to contextualise our research. Specifically, in Chapter §2 we introduce concepts related to business processes and we delve into the management of business processes and its performance. We describe process performance indicators and present in detail PPINOT, an approach for defining process performance indicators that is used as the basis for our overall proposal. In Chapter §3 we focus on the flexibility of business processes, On the one hand, different types of business processes that require more flexibility than traditional structured business processes are described. Business Process Variability is addressed in Section §3.2; and KiPs and an ontology for defining KiPs are described in Section §3.3. On the other hand, Section §3.4 is focused on decisions, a particular point of interest within business processes. Decisions are related to business process performance and we also address the performance of decisions themselves.

Part III: OUR PROPOSAL. This part is the core of this dissertation and is organised in four chapters as follows. In Chapter §4, we describe how reuse and abstraction can be applied in the performance perspective of business processes with the aim of reducing drawbacks derived from redundancy identified in PPI definitions. In Chapter §5, we identify, classify and describe several forms in which the variability is reflected in the performance perspective of business processes. In Chapter §6, we aim at improving the performance management in Knowledge-intensive processes. And finally, in Chapter §7, we identify and analyse the relationship between decisions and the performance of business processes from three different perspectives.

Part IV: FINAL REMARKS. Chapter §8 concludes this thesis, provides a definition of limitations identified, highlights some future work directions and summarises publications derived from the approaches presented in this manuscript.

Part V: APPENDICES. This part consists of five appendices. Appendix §A describes the SCOR model, a reference model that defines general processes and performance measures that can be applied to several organisations. This reference model is used in several scenarios described in this thesis. Appendix §B provides examples of using extensions of PPINOT taking into account abstraction concepts with the aim of facilitating reuse of PPI definitions. Appendix §C contains descriptions and examples of the extension of Visual PPINOT and shows how to integrate this extension with two well-known variability business process modelling languages used for modelling business processes reflecting their variability. Appendix §D presents a set of PPIs modelled using the KiPPINOT ontology. This PPIs are used in a case study detailed in Section §6.5. Finally, Appendix §E describes a preliminary analysis for measuring performance in CMMN cases, which can be considered unstructured processes.

Part II
BACKGROUND INFORMATION

Business Processes and Performance

"If you can't describe what you are doing as a process, you don't know what you are doing."

William Edwards Deming (1900-1993),
American statistician and professor

Business processes involve a set of inter-related events, activities and decision points that are related between them to achieve a desired business goal. Measurement of process performance is of great importance in order to identify whether the business objectives set are being achieved and to identify possible areas for improvement within the organisation. In this chapter, we introduce general concepts related to business processes and their performance. Section §2.1 introduces the chapter. Section §2.2 provides general concepts related to business processes. The business process management is described in Section §2.3. Concepts related to business process performance are explained in Section §2.4. Process Performance Indicators are presented in Section §2.5 as a mechanism for measuring process performance. Section §2.6 introduces PPINOT, an alternative to define PPIs and we also present different alternatives based on it to model PPIs. Section §2.7 describes reuse as a technique to improve the modelling of business processes. Finally, Section §2.8 summarises the chapter.

2.1 Introduction

Business processes are considered key instrument to organising the set of activities that are carried out within a company with the aim of providing a result to the market [165]. They can be defined as a collections of activities that uses inputs to creates outputs that are of value to the customer [64]; however, this definition can be considered limited because it does not consider the relationship between the activities as other authors like Davenport [26] does.

As stated by van der Aalst et al. [161], "*Business Process Management (BPM)*, intends to support business processes using methods, techniques, and software to design, enact, control, and analyse operational processes involving humans, organisations, applications, documents and other sources of information". It is gaining increasing interest from both academia and industry and it can be seen as a principle to manage business processes.

Business processes need to be measured so as to evaluate and continuously improve their performance [101]. The measurement and evaluation of business process performance is considered within the process Business Process Management and within the *Business Process Management Lifecycle*.

The goal of this chapter is to provide an overview of the major concepts related to business processes, business process management and process performance management, focusing on those aspects that are most interesting and useful for the purpose of this dissertation.

2.2 Business Processes

One of the simplest definitions of *business processes* is given by Hammer and Champy in [64], which describes a business process as a collection of activities that uses one or more inputs and generate outputs that are of value to the customer or the market. A more detailed definition is provided by Davenport [26], who considers relationships and restrictions between the collection of activities that make up a business process. Davenport defines a business process as "a specific ordering of work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs" and claims that "business process have customers (internal or external) and they cross organizational boundaries". In the proposal of Dumas et al. [41], a *business process* can be defined "as a collection of inter-related events, activities and decision points that involve a number of actors and objects, and that collectively lead to an outcome that is of value to at least one customer". And Weske, in [165], includes in its definition the relationship of several elements with an environment and organisation goals and states that a business process "consists of a set of activities that are performed in coordination in an organizational and technical environment. These activities jointly realize a business goal. Each business process is enacted by a single organization, but it may interact with business processes performed by other organizations."

The basis of business process management is the explicit representation of business processes [28, 165]. This representation may help discover weaknesses related to the order in which the activities are carried out or any other issue involved in business process execution. In this sense, according to Weske [165], business processes can be identified, reviewed, validated, and represented by *business process models*. They are considered the main artefacts for implementing business processes. He claims that a *business process model* "consists of a set of activity models and execution constraints between them". According to Decker [28], "business process models describe how a business goal is achieved through a number of activities".

A business process can be executed several times in an organisation, where each execution is called a *business process instance*. As stated in [165], a *business process instance* "represents a concrete case in the operational business of a company, consisting of activity instances" and each business process model acts as a blueprint for a set of business process instances.

In relation to the representation of business process models, Weske [165] also claims that "explicit business process models expressed in a graphical notation facilitate communication about these processes, so that different stakeholders can communicate efficiently, and refine and improve them". There are several proposals for modelling business process models, such as the Unified Modelling Language (UML) Activity Diagrams, the Event-driven Process Chain (EPC) and the Business Process Model and Notation (BPMN). They are usually called Business Process Modelling Languages (BPML) and have in common the support for specifying a process control flow, defining activities, decision points with alternative paths of execution, exception handling, event handling and additional rules and constraints [159].

UML [122] is a language focused on the development of software systems that has as its objective "to provide system architects, software engineers, and software developers with tools for analysis, design, and implementation of software-based systems as well as for modeling business and similar processes". It provides several types of diagrams that can be used in conjunction with activity diagrams (class diagrams, component diagrams, sequence diagrams and use case diagrams between others). In activity diagrams, the control flow requirements of processes are capture by depicting what activities are performed in what order and under what conditions. Activities are triggered by events an also can generate new events. State chart diagrams can be use to describe these actions.

EPCs [165] are part of the ARIS (Architecture of Integrated Information Systems) framework, a holistic modelling approach that defines a number of views which are similar to the dimensions previously defined for business processes. An EPC defines the control flow of business processes in terms of a sequence of events and functions [146]. Functions perform some business activities when they are triggered by events. Functions produce events while they carry out those activities.

The *Business Process Model and Notation (BPMN)* is considered the *de facto* standard for business process modelling. As stated in its specification [119], "the primary goal of BPMN is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes". A business process is depicted in BPMN as a business process diagram by means of a set of elements, those are: Collaboration and Process diagram elements, including all Task types, embedded Sub-Processes, CallActivity, all Gateway types, all Event types (Start, Intermediate, and End), Lane, Participants, Data Object (including DataInput and DataOutput), Message, Group, Text Annotation, Sequence Flow (including conditional and default flows), Message Flow, Conversations (limited to grouping Message Flow, and associating correlations), Correlation, and Association (including Compensation Association). The set also includes markers (Loop, Multi-Instance, Transaction, Compensation) for Tasks and embedded Sub-Processes. BPMN models are effective for the representation of predefined, fully specified and repeatable business processes.

A *business process* involves a set of elements that must be taken into account in its design and modelling, beyond the activities that make it up and the order in which they are executed. Proposals such as [28, 41, 42, 165] call these elements as perspectives or dimensions of business processes. Those proposals identify 5 perspectives of business processes that are described below.

- The *functional perspective* describes all activities to be performed in a business process.
- The *control flow perspective* indicates when activities and events should occur and specifies the order in which activities of a process must be performed. It is also known as *behavioural dimension*.
- The *resource perspective* focuses on who or what performs which activity: people, roles, organisational units, for example. According to [41] a resource is "a generic term to refer to anyone or anything involved in the performance of a process activity." This proposal distinguishes 3 types of resources: a process participant, such as an individual person; a software system, such as a software application; and equipment, such as a printer. This perspective is also known as *organisational dimension*.
- The *data perspective*, also known as *informational dimension*, specifies which information artefacts such as business documents or files are consumed to perform an activity and which of those artefacts are produced as a result of performing an activity.

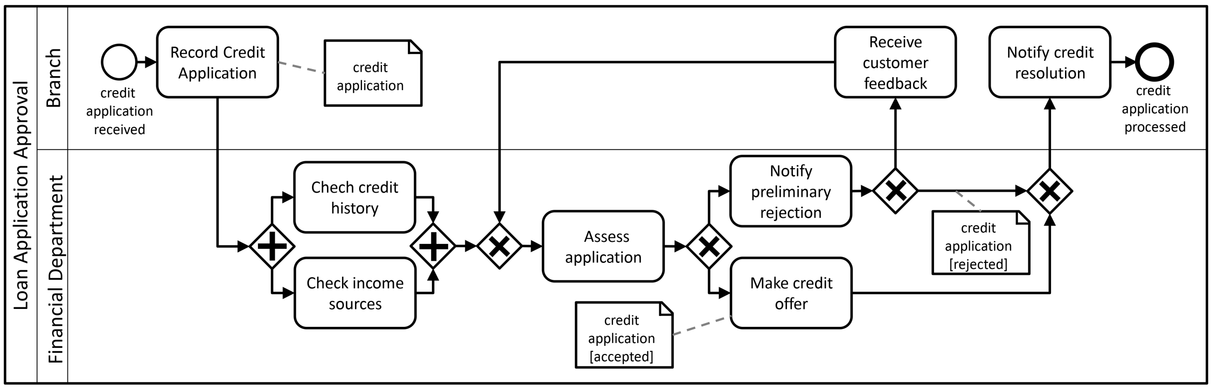

Loan Application Approval
Branch
Financial Department
credit application received
Record Credit Application
credit application
Receive customer feedback
Notify credit resolution
credit application processed
Chech credit history
Check income sources
Assess application
Notify preliminary rejection
Make credit offer
credit application [accepted]
credit application [rejected]

Figure 2.1: Loan application approval process modelled using BPMN

- The *technical perspective* concerns tools or machines that might be available for supporting the activities.

A loan application approval process is shown in Figure §2.1 as a business process model example. The model was created using BPMN. In this model, the functional perspective is identified by means of the activities depicted in the model; control flow dependencies between business process activities are represented by means of arrows that connect different activities; the resource perspective is slightly approached by means of the two lanes that indicate the roles that perform different activities; finally, data perspective is addressed y means of the data objects modelled (credit application).

2.3 Business Processes Management Lifecycle

The *Business Process Management (BPM)* can be defined as "supporting business processes using methods, techniques, and software to design, enact, control, and analyze operational processes involving humans, organizations, applications, documents and other sources of information" [161]. Weske [165] states that *Business Process Management* "includes concepts, methods, and techniques to support the design, administration, configuration, enactment, and analysis of business processes". The BPM can help to identify possible areas for improvement, but for this purpose, it is fundamental to measure the processes performance and identify whether the organisational goals of the company are being reached.

The BPM can be described by means of the *Business Process Management Lifecycle*. We based on the proposal of [165], in which the Business Process Management Lifecycle consists of four phases related to each other that are organised in a cyclical structure that shows their logical dependencies. This representation is shown in Figure §2.2 and each phase is described below.

- *Design and Analysis*: Design includes activities related to the identification and modelling of business processes. Analysis is related to the validation, simulation and verification of models.

- *Configuration*: This stage can be done in different ways. The implementation of a new system is not mandatory, because a business process can be realised without any support. If a software system is required, an implementation platform is chosen during this phase, with the aim of considering interactions of employees with the system and its integration with existing software systems. After configuration of systems, the implementation of the business process needs to be tested.

- *Enactment*: This phase encompasses the actual run time of the business process. "Process enactment needs to cater to a correct process orchestration, guaranteeing that the process activities are performed according to the execution constraints specified in the process model." In this stage, the monitoring of business processes is important to provide information about the status of its running instances. Valuable information can be gathered and stored from this monitoring phase related to the business process execution, such as times of activities performed.

- *Evaluation*: Information collected in previous stages is used it in this stage, with the aim of assessing and improving business process models and their implementations. The identification of business process models quality and the adequacy of the execution environment are performed in this stage.

In this dissertation, we focus on the performance of business processes. Business Process Performance is considered as part of the Business Process Management Lifecycle. This relationship is described in following sections.

2.4 Business Processes and their Performance

Performance management is defined as the process companies use to manage their performance in line with their strategies and objectives, with the aim of providing a control system, where their strategies are deployed to all business processes, activities, tasks and personnel; and obtaining feedback to enable appropriate management decisions [13].

Neely et al. [114] define performance measurement as "the process of quantifying action, where measurement is the process of quantification and action leads to performance". They also affirm that performance measurement is closely related to efficiency and effectiveness concepts. The former one indicates whether the requirements of customers are met and the latter "is a measure of how economically the firm's resources are utilized when providing a given level of customer satisfaction". Derived from these

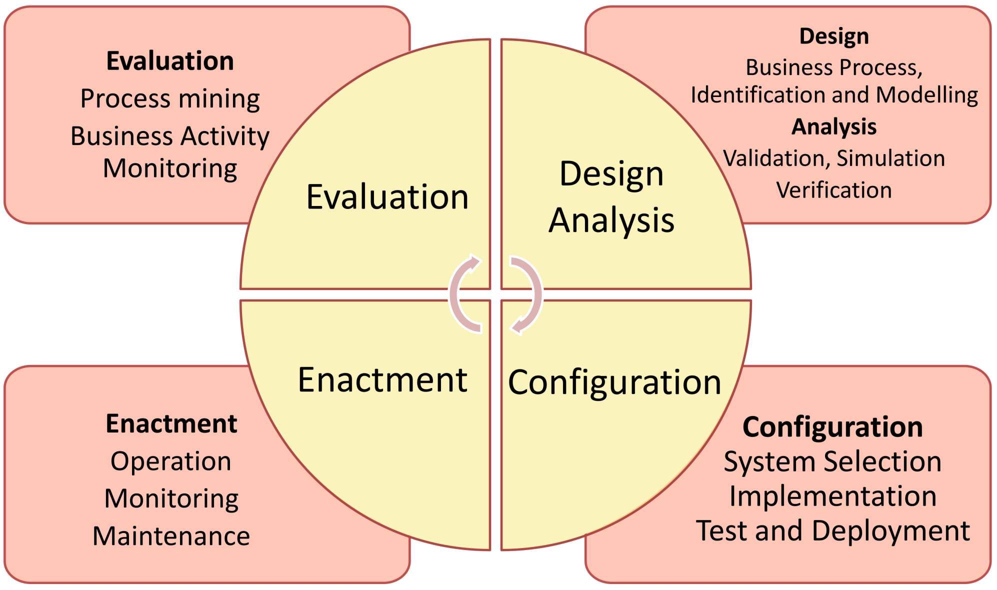


Figure 2.2: Business process management lifecycle as described by Weske in [165]

assertions, [114] also defines the performance measurement "as the process of quantifying the efficiency and effectiveness of action".

In the context of business processes, the *Business Process Performance Management* (or *Process Performance Management, PPM*) "provides detailed understanding of how a process can be designed and redesigned to improve the performance. It starts with identifying an opportunity for improvement and ends with an opportunity for continuous improvement" [100]. *Process Performance Management* is considered an essential part of the business process management that focuses on process of companies and "comprises the planning, monitoring, and control of process performance" [14]. The improvement of process performance is related to measuring the process performance and modifying that process with the aim of increasing the output performance [100]. For this purpose, process performance indicators are used in conjunction with other process support and analysis tools in order to organise those processes more efficiently [67].

Process Performance Management can be seen as the intersection between the *Business Process Management* and the *Performance Management*, as shown in Figure §2.3. For Business Process Management and Performance Management a number of phases or stages have been identified, including design/planning or monitoring. Although for both of them each of their stages has a different approach, we can say those stages, such as design/analysis, configuration, modelling, etc., are derived and applied from the point of view of Process Performance Management.

We can say that the performance of business processes is evaluated and monitored with the aim of identifying whether strategic and operational goals are being achieved in terms of efficiency and effectiveness. Over time, different techniques have been de-

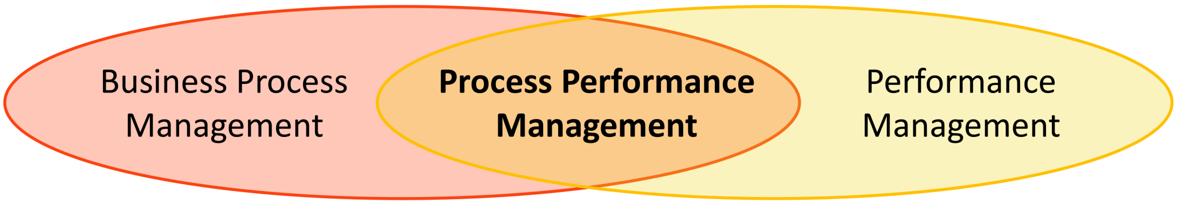


Figure 2.3: Relationship between Business Process Management and Performance Management

veloped with the aim of managing this performance. Nevertheless, the performance measurement of business processes is usually done through the identification, definition, computation and evaluation of process performance indicators [31, 130, 151].

2.5 Process Performance Indicators

The measurement of the performance of business processes, as part of the process performance management, is an active research field in management science, which has gained interest in both academia and businesses [130]. Much work has been performed on the identification and classification of key performance indicators in general settings [78] and those relevant for specific domains such as logistics, production, and supply chains [16, 21, 83, 158].

Process Performance Indicators (PPIs) are quantifiable metrics that provide useful and valuable information for the decision making and for the identification of possible areas for improvement in business processes. PPIs allow the evaluation of efficiency and effectiveness of business processes and can be measured using data that is generated within the process flow [31]. Directly or indirectly, PPIs are usually defined by means of a set of attributes that specify relevant information to establish what and how to measure the performance of business processes [31, 130]. Although there is no a general consensus on the attributes that best represent a PPI, the most relevant and recurrent are: the attributes required to identify the PPI, such as a name, an identification code, a general description, or other similar; a *process* in which the PPI is defined; a set of *goals* indicating the relevance of the PPI; a *measure definition* that specifies how to calculate the PPI; a *target* value to be reached indicating the fulfilment of the previously defined goals; the *scope*, described as the subset of instances to be considered to calculate the PPI value, and the *human resources* involved.

There are several ways to specify and define PPIs: in an informal way using natural language [166], using technical specifications [55], defining or extending metamodels and ontologies that allow a more formal definition of performance indicators and their relationship with the business process [31, 81, 108, 125, 166], in a structured way using tables [144] or templates [32, 130], or by means of graphical notations [34, 52]. In addition, [163] identifies weaknesses and inadequacies concerning the definition of PPIs in

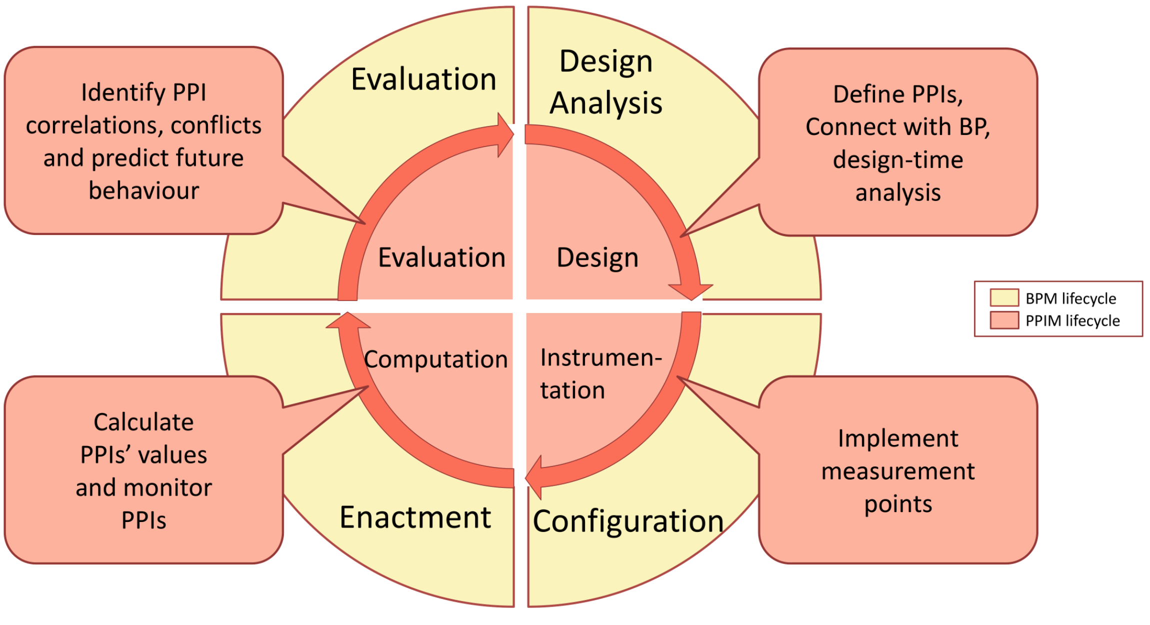

Evaluation
Design
Analysis
Identify PPI correlations, conflicts and predict future behaviour
Define PPIs, Connect with BP, design-time analysis
Evaluation
Design
Computation
Instrumen-tation
Calculate PPIs' values and monitor PPIs
Implement measurement points
Enactment
Configuration
BPM lifecycle
PPIM lifecycle

Figure 2.4: PPI management lifecycle integrated into the Business Process Management Lifecycle. Taken from [30]

a structured literature review about performance measurement in the business process field. These alternatives differ from each other in their expressiveness, i.e. the different types of PPIs that can be defined, and their features to directly support monitoring.

In Section §2.3, the business processes management was described by means of a lifecycle associated to business processes. In relation to PPIs, proposals such as [30, 85] describe a lifecycle for the management of PPIs and relate both lifecycles by means of their different phases. Figure §2.4, taken from [30], shows the relationship between the business process management lifecycle and the PPI management lifecycle. The Design and Analysis, Configuration, Enactment and Evaluation phases make up the business process management lifecycle, which are related to the Design, Instrumentation, Enactment and Evaluation phases established in [30] as the PPI management lifecycle phases, respectively.

In this dissertation we focused on the improvement of the first stage of the PPI management lifecycle, where PPIs should be design, analysed and modelled. In this phases, the PPI should be defined by means of a set of attributes and its structure should be specified, clearly describing its relationship with the business process.

2.6 PPINOT

In this dissertation we focus on the modelling perspective of process performance management, specifically on the modelling of process performance indicators. As we mentioned in Section §2.5, there are several proposals for defining PPIs using different techniques: natural language [166], technical specifications [55], tables [144], templates [32, 130] or by means of a graphical notations [34, 52]. Other efforts have been aimed at defining or extending metamodels and ontologies that allow a more formal definition of performance indicators and their relationship with the business process. For example, Pedrinaci et al. [125] presents SENTINEL (SEmaNTic busINess procEsses monitoring tooL), a tool that contemplates two modules: one is the Metrics Computation Engine that "is in charge of supporting the automated computation of general purpose as well as user defined metrics" and the second, the Deviations Detection Engine, that "aims to support the automated detection of process deviations". In [108], Momm et al. propose a metamodel for the specification of the PPI monitoring, an extension of the BPMN metamodel for modelling the instrumentation for the monitoring, and a methodology for an automated generation of this instrumentation. In [81], the BPMN and EPC metamodels are extended to define business process goals and performance measures. González et al. [55] propose a domain specific language called the MMC-BPM language, to complement business process models with monitoring, measurement and control concerns.

In this dissertation, we selected PPINOT, a metamodel for defining and modelling PPIs. This approach has been previously used in various scenarios and it fulfils a set of desirable characteristics such as: high expressiveness, it allows the definition of PPIs in an unambiguous and complete way, it facilitates traceability between business process elements and PPIs, it promotes the fulfilment of SMART criteria (specific, measurable, achievable, relevant and time bounded) and it is independent of the language used to model the business process. In addition, PPINOT is able to be extended and adapted to new contexts and requirements.

The PPINOT Metamodel is graphically represented in the UML diagram shown in Figure §2.5. White boxes represent PPINOT classes and white boxes with underlined names represent elements of the *Abstract Business Process Modelling Language (ABPML)* that represents business process elements and with which the PPINOT elements are related. This ABPML allows PPINOT to be used with more than one Business Process Modelling Language (BPML), although it has traditionally been used with BPMN [119]. In PPINOT a PPI is defined by means of a set of attributes, which are described below:

- `identifier:String`. A PPI must be uniquely identified. In PPINOT this attribute is usually represented by a number preceded by the text *PPI*.

- `name:String`. This attribute provides a descriptive name for every PPI.

- `goals:String[0..*]`. This attribute is an expression represented in natural languages to highlight the relevance of the PPI. It allows the user to establish the strategic or operational goals that the PPI is related to.

- `responsible:HumanResoure`. It represents the human resource - a person, a role, a department or an organisation - in charge of the PPI.
- `informed:HumanResoure[0..*]`. This attribute refers to a human resource or a set of them: people, roles, departments and organisations, which are informed about the PPI.
- `comments:String`. Other information about the PPI that cannot be fitted in previous fields can be recorded here.
- `target:Target`. This attribute represents a target value to be reached by the PPI indicating the consecution of the defined goals. In PPINOT there are three different ways to define a target value. A *simple target* is used to specify the lower and/or upper bound that make up the range within which the PPI value should be. For example, for a PPI that measures "*the average time required for the execution of the activity Asses application*" a simple target could be defined as "less than 2 days". The *composed target* allows the definition of several target values or ranges. For example, indicating that the expected value for the same PPI should be "more than 2 hours and less than 24 hours" for urgent applications. Finally, the *custom target* allows the definition of a restriction that the PPI value must fulfil. Figure §2.6 details the excerpt of the metamodel related to the target attribute.
- `scope:Filter`. By means of this attribute it is possible to define a subset of instances of the associated process that must be considered to compute the PPI value. Several filters can be defined attending to conditions on the number of instances, temporal aspects and the process instance state. For example, for the PPI introduced in the previous item, the scope can be defined to consider all instances of the business process or a subset of them, such as a instances generated in "a specific period of time or holidays, or the last *n* instances of the process". Figure §2.7 shows the PPINOT metamodel excerpt with all classes and restrictions that allows the definition of filters.
- `relatedTo:Process`. This attribute refers to the process for which the PPI is defined. This process is external to the PPINOT metamodel.
- `definition:MeasureDefinition`. This attribute specifies how the PPI should be computed. The `measure definition` is a complex attribute that can be represented by a variety of measures:
  - *`Base measures`* is the first type of measures that represents a single-instance measure that measures values of time, count, state conditions or data: *Time measure* measures the duration of time between two time instants, such as "the duration between the time instant when activity *Assess application* changes to state active and the time instant when activity *Notify credit resolution* changes to state completed"; *Count measure* measures the number of times something happens, such as "the number of times activity *Assess application*

changes to state completed"; *State condition* measures the fulfilment of certain condition in a process instance, such as "the fulfilment of the condition applicationState=rejected over the data object *Credit application*"; and *Data measure* measures the value of a certain part of a data object. For example, if the PPI definition corresponds to "type of clients that present Credit applications", the measure could be defined as "the value of client type of *Credit application*".

– `Aggregated measure` is the second type, which apply an aggregation function - average, maximum, minimum, sum, etc. - to the set of defined values of a measure corresponding to the process instance in PPI scope. An example of this measure definition is "the sum of the number of times event *credit application received* is triggered".

– Finally, the third type is `Derived measure`, which represents either a single-instance or a multi-instance measure whose value is obtained by calculating a mathematical function over other measures. For example, a derived measure can be defined as "the mathematical function (a/b)*100 where a is the number of times data object *Credit application* is in state accepted and b is the number of times data object *Credit application* is in state registered".

Measures are connected with processes by means of `Conditions` that indicate how and when to take values from the process, and `DataContentSelections` to obtain an attribute of a data object. `Time measure` requires two conditions, *from* and *to*, to indicate the start and end point of for measuring; `Count measure` needs a *when* condition that indicates the point when something happens and should be measured; `State condition` measure uses a *meets* condition to indicate the condition whose fulfilment is being measured and a `Data measure` needs a *measuresData* condition to select the part of the data object that is being measured.

On the basis of the PPINOT metamodel, two notations were proposed. The first one is the graphical notation *Visual PPINOT* [34] developed with the aim of reducing the visual gap between the business process models and models of PPIs. *PPINOT Templates and Linguistic Patterns* [32] constitute the second mechanism to define PPIs, that seeks to define PPIs in a fixed way being understandable for different types of users. Both notations are described in detail the following subsections.

2.6.1 PPINOT Templates and Linguistic Patterns

PPINOT Templates and Linguistic Patterns were developed on the basis of the PPINOT Metamodel. This notation helps to structure the PPI information in a fixed form thus reducing ambiguity and serving as a guide to avoid missing relevant information. Table §2.1 shows the template for a PPI definition taken from [32]. Each row in the table describes a PPI attribute. Values of attributes should follow this specification: words between "<" and ">" are placeholders for either literals (lower case) or

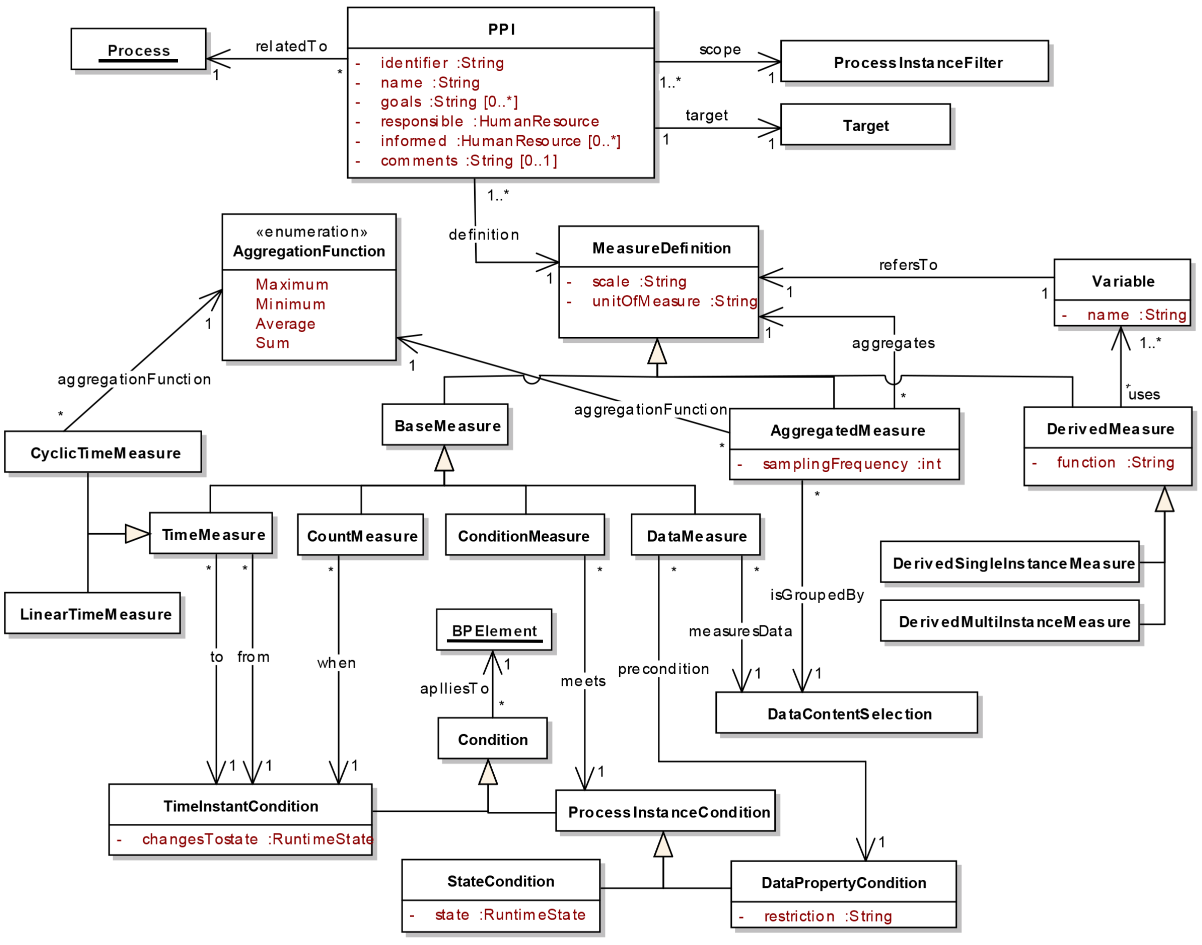


Figure 2.5: PPINOT Metamodel (Taken from [31])

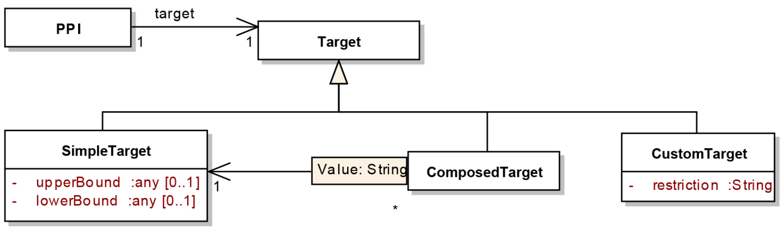


Figure 2.6: PPINOT Metamodel - Target definition (Taken from [31])

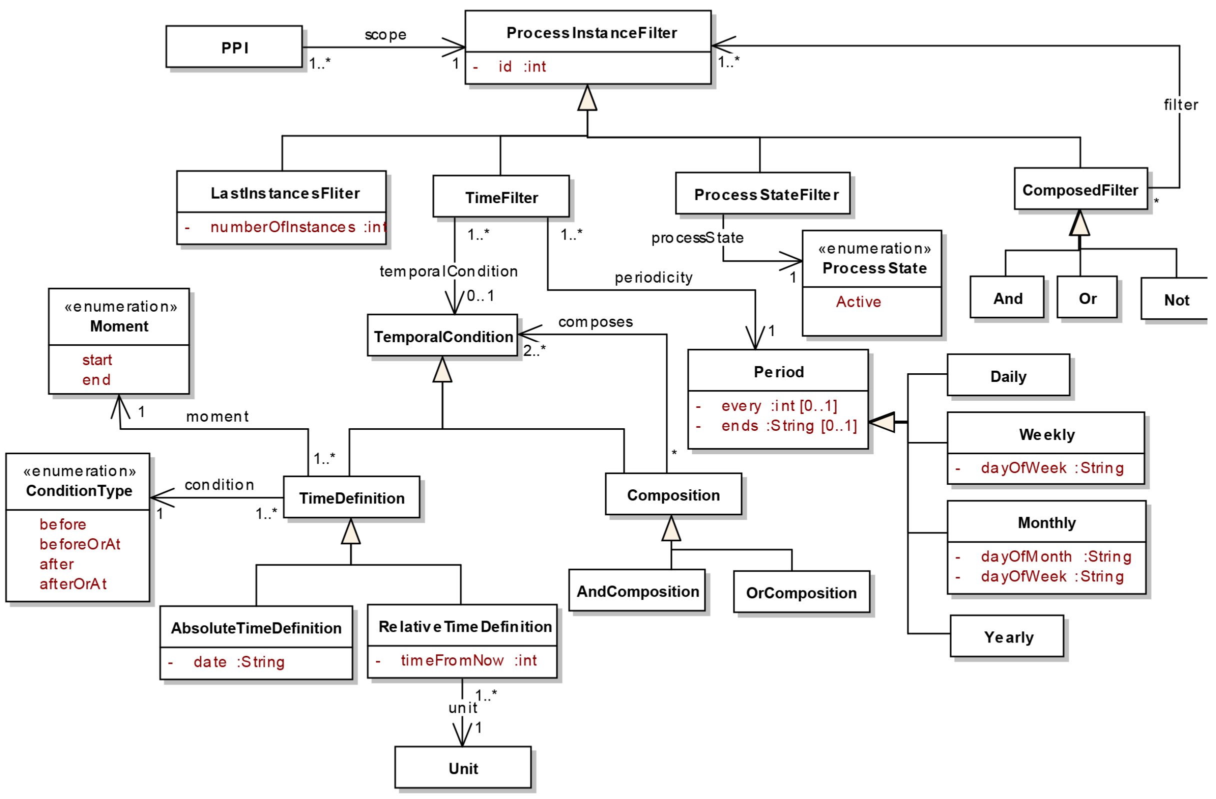


Figure 2.7: PPINOT Metamodel - SCOPE - filter definition (Taken from [31])

linguistic patterns (upper case first letter), words between "[" and "]" are optional and words between "{" and "}" and separated by "|" are one-only option.

Linguistic Patterns allow user-friendly definitions of PPIs. Since a PPI can be defined using different types of measures (time, count, etc.), a set of patterns were identified to describe each PPINOT measure. Table §2.2 gathers PPINOT Linguistic Patterns.

On the basis of the information presented in Tables §2.1 and §2.2, we present an example of a PPI defined using PPINOT templates and linguistic patterns in Table §2.3. The PPI "Average time of assess Credit Application" (PPI-1) is associated with the process "Loan Application Approval" with the aim of reducing the credit application time to response (Business Goal). This PPI is defined using two PPINOT measures: an aggregated measure to define the aggregated function "average" and the time measure to define the starting and ending point which determine the time period measured by the PPI during the execution of the process. PPI-1 is calculated using all process instances (Scope). Information is taken from event logs generated by information systems in the organisation. Financial Analyst is the responsible for the execution of the PPI and Financial Manager must be informed of the results obtained. No additional comments are included in this PPI definition.

Table 2.1: PPINOT Template in PPI definitions.

PPI- $< id >$	$<$***PPI descriptive name***$>$
Process	$<$ *process name (process id)* $>$
Goals	$<$ *strategic or operational goals the PPI is related to* $>$
Measure Definition	The PPI value is calculated as $<$ *Measure*$>$
Target	The PPI value $<$ *Target constraint* $>$
Scope	The process instance considered for this PPI are: • all • [those in] $<$*Scope(S-x)*$>$
Source	$<$ *source from which the PPI measure can be obtained* $>$
Responsible	$\{ < role > \| < department > \| < organisation > \| < person > \}$
Informed	$\{ < role > \| < department > \| < organisation > \| < person > \}$
Comments	$<$ *additional comments about the PPI* $>$

2.6.2 PPINOT Graphical Notation

Visual PPINOT is a graphical notation for defining process performance indicators based on the PPINOT Metamodel. This notation allows the definition of PPIs together with business process models, but regardless of the business process modelling language used to model the business process. Figure §2.8 gathers the elements of the notation.

The PPI is depicted by means of a rectangle with a gauge icon on its upper left corner. This representation of the PPI includes the attributes: identifier, displayed centred at the top; the target value, which is represented as a bullseye and the scope value represented as a grid, are displayed in an optional gray bottom compartment. Finally, the measure defining the PPI is displayed inside the rectangle.

The three types of PPINOT measures (base, aggregated and derived) are represented as short rules with their names underneath, but depending on the type of measures different characteristics are added. Base measures are differentiated by an icon in the upper left corner of measure: a set of numbers within an ellipse for the count measure, a hourglass for the time measure, a page with the corner turned for the data measure, and an check symbol within an ellipse for the state condition. Symbols depicted inside a State condition measure represent the aggregation function to be used: number of process instances, percentage of process instances, all process instances, at least one process instance or no process instance. These aggregation functions are different from other measure types. Aggregated measures are depicted three stacked base measure icons with an aggregation function inside: average (AVG), summation (SUM), maximum (MAX), etc. They are connected to the single-instance measure being aggregated using aggregates connectors, depicted as solid lines starting with a white diamond and labelled with *aggregates*. Optionally, aggregated values can be grouped by some data object attribute, depicted as a dashed line starting with a white diamond and labelled

Table 2.2: Linguistic Patterns for Measure Definitions in PPINOT

Measure	**Linguistic Patterns**
TimeMeasure ::=	*LinearTimeMesure* | *CycleTimeMeasure*
LinearTimeMesure ::=	*the duration between the [first] time instant[s] when* $< Event_1 >$ *and [the last time instant] when* $< Event_2 >$
CycleTimeMeasure ::=	*the { total | maximum | minimum | average | ...} duration between the pairs of time instants when* $< Event_1 >$ *and* $< Event_2 >$
Event ::=	*<BP element type>*[a] *<BP element name>*[b] *becomes <BP element state>*[c] | *<BP event name>*[d] *is triggered*
CountMeasure ::=	The number of times $<$ *Event* $>$
ConditionMeasure ::=	*<StateCondM>*|*<DataPropertyCondM>*
StateCondM ::=	*<BP element type>*[a] *<BP element name>*[b] *[that] {[is] [not] currently | has [not] finished} [in state] <BP element state>*[c]
DataPropertyCondM ::=	*[<data object state>] <data object name> that satisfies: <condition on data object properties>*
DataMeasure ::=	*the value of [property] <data object property name> of [data object] <data object name>*
DeriverdMeasure ::=	*the function <expression over* $x_1...x_n$ *>, where* $\{< x_i >$ *is* $< MeasureForDer_i >\}_{i=1..n}$
MeasureForDer ::=	*TimeMeasure | CountMeasure | ConditionMeasure | DataMeasure | AggregatedMeasure*
AggregatedMeasure ::=	*the { sum | maximum | minimum | average | ... } of <MeasureForAgg> [grouped by [property] <data object property name>of [data object] <data object name>]*
MeasureForAgg ::=	*TimeMeasure | CountMeasure | ConditionMeasure | DataMeasure | DerivedMeasure*

[a] *<BP element type>*: One of the different types of elements of the business process referred. In the case of BPMN 2.0, they can be: *activity, data object, event* and *process*.
[b] *<BP element name>*: The name of one of the existing BP elements in the process referenced in the PPI.
[c] *<BP element state>*: Possible states for each type of BP element. For instance, in BPMN 2.0, the possible states for activities are *ready, active, withdrawn, completing, completed, failing, failed, terminating, terminated, compensating* and *compensated*.
[d] *<BP event name>*: The name of one of the existing events in the process referenced in the PPI.

Table 2.3: Example of PPI definition

PPI-1	**Average time of assess Credit Application**
Process	Loan Application Approval
Goals	Goal01: Reduce Credit application time to response
Measure Definition	The PPI value is calculated as *the average of the duration between the time instant when activity Assess application becomes active and when activity Assess application becomes finished.*
Target	The PPI value *must be less than or equal to two working days*
Scope	The process instance considered for this PPI are all
Source	Event logs
Responsible	Financial Analyst
Informed	Financial Manager
Comments	-

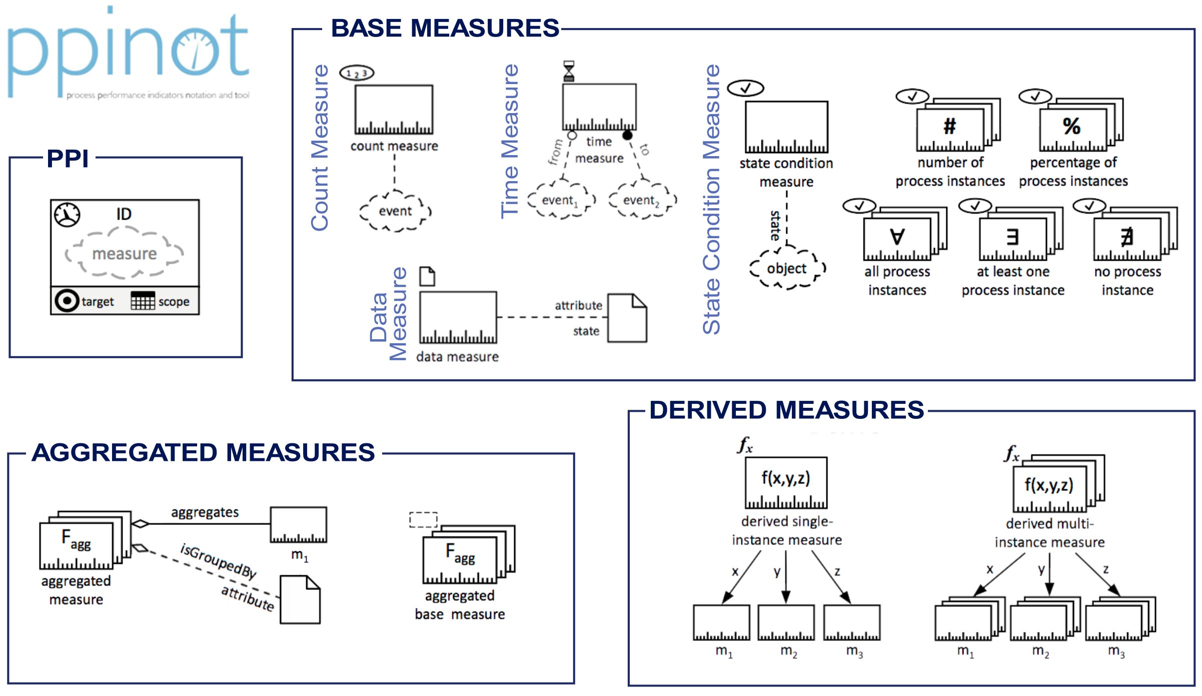


Figure 2.8: PPINOT Graphical Notation

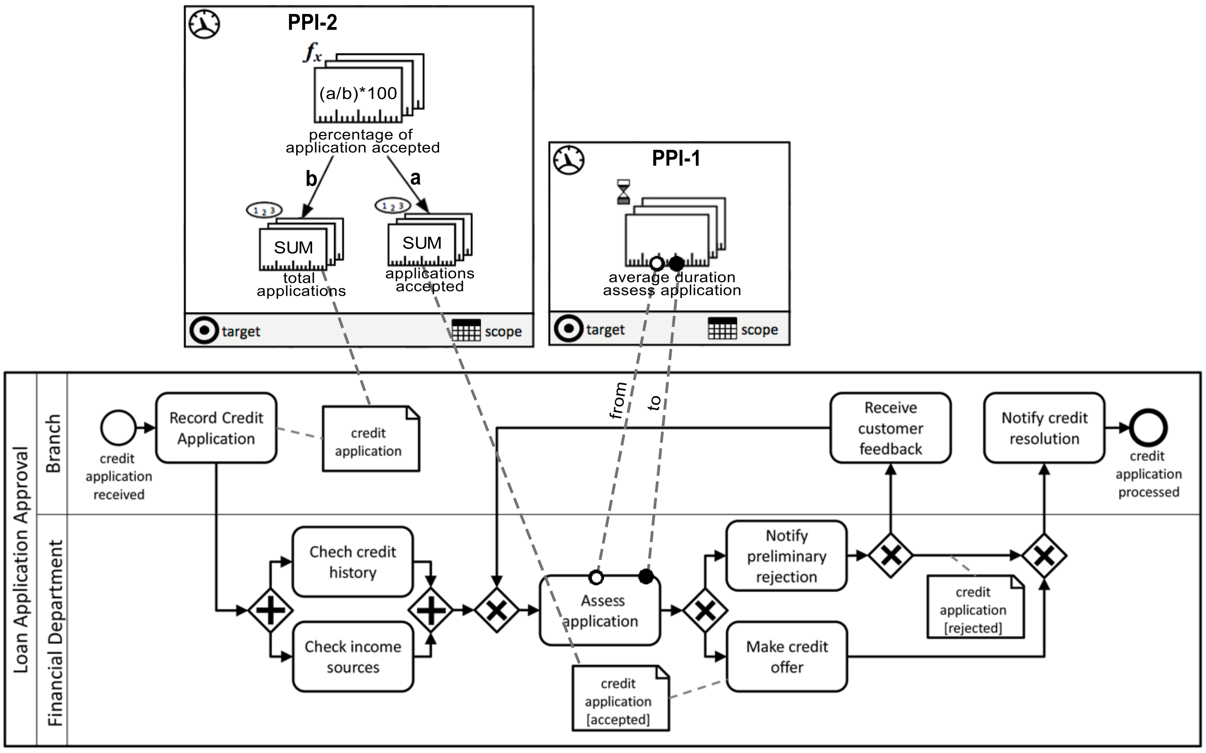

PPI-2
fx
(a/b)*100
percentage of application accepted
b
a
SUM
total applications
SUM
applications accepted
target
scope
PPI-1
average duration assess application
target
scope
from
to
Loan Application Approval
Branch
Financial Department
credit application received
Record Credit Application
credit application
Receive customer feedback
Notify credit resolution
credit application processed
Chech credit history
Check income sources
Assess application
Notify preliminary rejection
credit application [rejected]
Make credit offer
credit application [accepted]

Figure 2.9: Examples of PPIs using the graphical notation of PPINOT

with *isGroupedBy*. Aggregated base measures can use a compact form with one icon only. Both representations can be found in Figure §2.8. Finally, *Derived measures* can be single- or multi-instance. Both types are represented by a function symbol (fx) on the upper left corner and by the expression of their derivation function inside the ruler icon that defines the measure. They differ from each other in that the multi-instance is represented by a three-stacked rule icon. All measures involved in a derived measure must be single- or multi-instance according to the derived measure being defined. *Uses* connectors are used to connect function variables to derived measures.

Figure §2.9 shows an example of two PPI definitions. The first one, *PPI-1* is the same PPI defined using PPINOT Templates (See Table §2.3). For this PPI we have used an aggregate measure over a measure of time, for which we use the abbreviated version of the measure, represented by the measure "average duration assess application". The *PPI-2* is defined by means of a derived measure (percentage of application accepted) that calculates "the percentage of credit application accepted". To do this, we use a multi-instance measure, because we are considering information from several process instances of the business process. This derived measure requires information from two other measures (total applications and application accepted), each of which is an aggregated measure applied over a count measure.

2.6.3 PPINOT Formal Definition

In this section, we present a new representation of the PPINOT metamodel, which has been developed with the aim of formally describe its components, the relationship between them and the restrictions on their use. This formal definition will be used in later sections to extend the metamodel with new features.

The PPINOT metamodel allows the definition of a *performance model* as a set of PPIs. In this section we provide a formal definition for that performance model and for the PPINOT metamodel concepts described from the beginning of this section.

In order to formally define a PPINOT performance model, we first need to formalise the concept of *Condition*, which is the link between the performance model and the other elements of the business process.

Definition 2.1 - Condition.
Let bp be a *business process*, $\mathcal{A}$ be a not empty set of *activities* for bp, $\mathcal{S}_{\mathcal{A}}$ be a set of *activity states* of $\mathcal{A}$, $\mathcal{D}$ be a finite set of *data objects* for the bp, $\mathcal{S}_{\mathcal{D}}$ be a finite set of *data object states* of $\mathcal{D}$, $\mathcal{A}_{\mathcal{D}}$ be a non-empty set of *data object attributes* of $\mathcal{D}$, $\mathcal{E}$ be a non-empty set of *events* for the bp, $\mathcal{S}_{\mathcal{E}}$ be a set of *event states* of $\mathcal{E}$. $\mathcal{C}_{bp} = \mathcal{A} \times \mathcal{S}_{\mathcal{A}} \cup D \times \mathcal{S}_{\mathcal{D}} \cup \mathcal{E} \times \mathcal{S}_{\mathcal{E}}$ is the set of all possible *Conditions* that can be defined over bp.

For example, a condition $\mathcal{C} = (Activity\ Assess\ Application, active)$ represents the moment when *Activity Assess Application* becomes *active* in a given running instance.

Now, a PPINOT performance model can be defined as follows.

Definition 2.2 - PPINOT Performance Model.
Let *bp* be a business process, $\mathcal{C}_{bp}$ be the set of all possible conditions defined over *bp*, $\mathcal{S}$ be the set of *scopes* that can be defined for a PPI, $\mathcal{T}$ be the set of targets that can be defined for a PPI, $\mathcal{HR}$ be the set of human resources that can be related to the PPI, $\mathcal{F}_{agg} = \{MIN, MAX, AVG, SUM, \ldots\}$ be a set of *aggregation functions*. A performance model *PM* over $\mathcal{S}$, $\mathcal{T}$, $\mathcal{HR}$, $\mathcal{C}_{bp}$ and $\mathcal{F}_{agg}$ is a tuple $PM = (P, M, L_P, L_M)$, where:

- P is the set of *process performance indicators* of a bp;
- $M = BM \cup AggM \cup DerM$ is a set of *measure definitions*, where:
  - $BM = TimeM \cup CountM \cup StateM \cup DataM$ is a finite set of *base measures*, where: *TimeM*, *CountM*, *StateM*, *DataM*, are the set of *time*, *count*, *state condition* and *data* measures defined by *PM*, respectively.
  - *AggM* is the set of *aggregated measures* defined by *PM*;
  - *DerM* is the set of *derived measures* defined by *PM*;

- $L_P = sco \cup tar \cup res \cup inf \cup mes$ is the set of links between a PPI $p \in P$ and its attributes, where:
  - $sco \subseteq P \times \mathcal{S}$ is the set of *scope links* assigned to each PPI;
  - $tar \subseteq P \times \mathcal{T}$ is the set of *target links* assigned to each PPI;
  - $res \subseteq P \times \mathcal{HR}$ is the set of human resource links to indicate the person *responsible* of the PPI;
  - $inf \subseteq P \times \mathcal{HR}$ is the set of human resource links to indicate the people *informed* about the PPI;
  - $mes \subseteq P \times M$ is the set of links with the *measure* that defines each PPI;
- $L_M = cond \cup data \cup agg \cup cyclic \cup uses \cup derfun$ is the set of links between *measure definitions* and its attributes, where:
  - $cond = from \cup to \cup when \cup meets$ is a set of links among measures and conditions, where:
    - $from \subseteq TimeM \times \mathcal{C}$ is the set of links to *time conditions*, *from*;
    - $to \subseteq TimeM \times \mathcal{C}$ is the set of links to *time conditions* of *to* type;
    - $when \subseteq CountM \times \mathcal{C}$ is the set of links to *time condition*, *when*;
    - $meets \subseteq StateM \times \mathcal{C}$ is the set of links to *state conditions*, *meets*;
  - $data \subseteq DataM \times \mathcal{D} \times \mathcal{S}_{\mathcal{D}} \times \mathcal{A}_{\mathcal{D}}$ is the set of links to *data* conditions;
  - $cyclic \subseteq TimeM \times \mathcal{F}_{agg}$;
  - $agg \subseteq AggM \times (BM \cup DerM) \times \mathcal{F}_{agg}$ is the set of functions to measure a set of process instances when an aggregated measure is used;
  - $uses \subseteq DerM \times M \times \mathbb{N}$ is the set of links between a derived measure and the set of measures involved with it;
  - $derfun \subseteq DerM \times F$ is the set of links between derived measures and its functions, where: F is the set of all possible functions that could be resolved using derived measures;

Given a connector link $lm \in L_M$, $\Pi_M(lm)$ represents the measure involved in lm and $type_M(lm) \in T_M$, where $T_M = \{from, to, when, meets, cyclic, data, agg, uses, derfun\}$ represents the type of the link. For instance, let $lm = (m_1, c_1) \in from$, $\Pi_M(lm) = m_1$ and $type_M(lm) = from$.

Similarly, given a connector link $lp \in L_P$, $\Pi_P(lp)$ represents the PPI where the attribute has been assigned and $type_P(lp) \in T_P$, where $T_P = \{sco, tar, res, inf, mes\}$ represents the type of the link. We also define $L_P[p,t]$ as the subset of L_P whose PPI is p and whose type is t, i.e., $L_P[p,t] = \{lp \in L_P \mid \Pi_P(lp) = p \wedge type_P(lp) = t\}$. Likewise, $L_M[m,t]$ is the subset of L_M whose measure definition is m and type is t, i.e., $L_M[m,t] = \{lm \in L_M \mid \Pi_M(lm) = m \wedge type_M(lm) = t\}$.

We can now define a syntactically correct PPINOT performance model PM. This is based on the metamodel specification introduced in [31] and displayed in Figure §2.5 We mainly specify restrictions about relationships of measuring elements and define link constraints between PPIs and its attributes and between measures and its connectors.

Definition 2.3 - Syntactically correct PPINOT performance model.
Let $PM = (P, M, L_P, L_M)$ be a performance model, PM is syntactically correct if it fulfills the following requirements:

1. There is at least one PPI p in the performance model $\mid P \mid > 0$.
2. Each PPI attribute can only have exactly one single value linked to the PPI, except for the *informed* attribute. $\forall p \in P, t \in T_P \setminus \{inf\}(\mid L_P[p,t] \mid = 1)$
3. Measures have at most one link for each possible type of link in L_M except for *uses*: $\forall m \in M, t \in T_M \setminus \{uses\}(\mid L_M[m,t] \mid \leq 1)$
4. Depending on its type, measures have at least one element of their links:
   - $\forall tm \in TimeM(\exists(tm, c_i) \in from \land \exists(tm, c_j) \in to)$
   - $\forall cm \in CountM(\exists(cm, c) \in when)$
   - $\forall sm \in StateM(\exists(sm, c) \in meets)$
   - $\forall dm \in DataM(\exists(dm, d, s, a) \in data)$
   - $\forall am \in AggM(\exists(am, m) \in agg)$
   - $\forall dm \in DerM(\exists(dm, f) \in derfun)$
   - $\forall dm \in DerM(\exists(d, m, x) \in uses)$
5. A derived measure cannot be related to more than one measure with the same identifier: $\forall(d, m_i, x) \in uses \neg\exists(d, m_j, y) \in uses\ (x = y \land m_i \neq m_j)$
6. The identifiers used for a derived measure should be sequential, which is ensured if the highest identifier is equal to the number of *uses* links for such derived measure: $\forall(dm, m_i, x) \in uses(x \leq \mid L_M[dm, uses] \mid)$.
7. For all $(d, f) \in derfun, f \in F$ must be a function defined over the Cartesian product of the set of all possible values of the set of measures linked to d $(\{m \in M \mid (d, m, x) \in uses\})$, ordered according to x

---

2.7 Reuse in Business Process Management

Organisations typically define a large number of business processes [36]. According to several authors [50, 99, 169], the modelling of large number of business processes can become a complex and error prone task, because this action may require a lot of time, as well as being able to generate large repositories with redundant information. To face this challenge, the *reuse* of certain parts of business process models is proposed as an alternative to reduce the time spent in creating a solution from the scratch, by reducing efforts, time and costs; and also to reduce the possibility of introducing errors during their management.

Several literature reviews relating reuse with business processes can be found. In [169], the authors claim that "the quality of a business process model has a direct effect on the business performance" and "the reuse can accelerate the design process and produce high quality solutions". They describe five artefacts found in literature for the reuse of knowledge in construction of business process models: workflow patterns, workflow activity patterns, action patterns, reference models and semantic business process patterns; and highlighted characteristics they consider to have an impact on the design effort and on the quality of the business process model generated by reuse, such as granularity, abstraction level (in terms of the number and complexity of modification operations needed to reuse the artefacts: addition, deletion, renaming, etc.), context of use and reuse guidance. The review presented in [4] identifies four types of modelling solutions related to the reuse in business processes: patterns, process discovery, declarative approaches and ontologies, which can be used together because some of them have similar or complementary characteristics. This review is considered by its authors to be the first step in proposing a methodology aimed at discovering process patterns that facilitates the reuse in the design and redesign of business processes.

In [36], a discussion about a variety of techniques for managing large collections of business processes is presented. A discussion about a variety of techniques for managing large collections of business processes is presented in [36]. In this review, *reuse* is one of the nine areas studied related to the managing of business process model collections. In [50], Fellmann et al. present a taxonomy for business process reuse is proposed. Empirical research and research where new assets are designed and suggested are the two main groups of the taxonomy. The latter category is subdivided into technical artefacts (architecture, framework or repository)) and methods, which in turn are subdivided into subcategories: abstraction, selection, specialisation and integration. Abstraction is the subcategory with the most references to approaches found in the literature, which include publications related to reference modelling, meta models or patterns. The proposal presented in [68] states that there are several examples of design and development of systems that focus on reuse of artefacts framed in different organisational contexts: code reuse, component reuse, reuse of industrial reference models such as SCOR, design patterns, etc. But it also claims that successful reuse depends on factors such as "the variability of the problem domain, the availability and applicability of a reusable design artefact, and the human cognitive activities that are performed when identifying and, most importantly, adapting an artefact for a present managerial challenge". A comparative study between two scenarios is conducted to test the impact of information granularity on the effectiveness of business process model reuse. Their experimental results show that there is a difference in one out of three defined modelling effectiveness variables.

Other specific proposals related to reuse in business processes are as follows. In [99], a framework based on a business process ontology is presented for the reuse of business process fragments. Authors define a process fragment as "a self-contained, coherent building block of a process model with a clear business meaning". The reuse of these process fragments reduces modelling time because avoid the modelling of the same process fragment several time. The process fragment reuse can be applied without modifications or by the process fragment adjustment to new contexts. In [124], authors propose the use of patterns for the modelling of business processes. To do this, they propose the extension of a methodology (previously described in [23]), which aims to support the modelling of business processes, as well as "the capture and record of analysis experiences, pattern evolvement and their reuse in future developments". In [170], a software component called Process Assembler is presented for the purpose of providing key functionality for business process reuse, such as adapt a business process to the case at hand or assess a business process as to whether or not it should be reused in the case at hand, etc.

As noted in several previous approaches and literature reviews, abstraction is a concept related to reuse in business processes. For this reason, below we comment on some works related to abstraction in general and to abstraction in business processes in particular. Abstraction is considered a natural mechanism to focus attention on some aspects and on relevant information in a particular moment, which may occurs in several ways, for example focusing on properties of perceived objects or focusing on actions on properties, and can be applied in different context [56]. According to [27], "a model abstraction is an operation that reduces the complexity of some aspect of a model" with the aim of increasing the comprehensibility of a large model or reducing the size of a model to facilitate its verification while retaining certain properties of interest.

From the point of view of business processes, abstraction has been a point of research and interest in recent years, which have been mainly oriented to reduce complexity in business process modelling. Business Process Model Abstraction has been addressed in several approaches such as [15, 43, 129, 152]. There is more than one classification of abstraction. Applicability and behaviour are the main classification factors used in [27]. The former is subdivided in horizontal and domain-specific abstraction; while the latter is subdivided in merger, aggregation, deletion and views. In the proposal of [84], the classification of abstraction includes: aggregation, generalisation and association.

In the proposal of Smirnov et al. [149], a formal framework for business process model abstraction is proposed. This approach proposes the reuse of information using cartographic generalisation that consists of selecting and representing information in a way that adapts to the scale of the display medium. This approach also provides a survey about the state of the art of business process model abstraction. The proposal of Polyvyanyy et al. [129] is also based on the idea of cartographic generalisation. The use of abstraction to present a readable, high-level view of a business process model, by showing aggregated activities and leaving out irrelevant details is described in [148].

A preliminary work based on a view-based framework is introduced in [157] to define concerns of the process model (orchestration, collaboration and information views) and uses semi-formalised models to capture a particular perspective of the business process by means of different views. Draheim [39] discusses the decomposition of control flow and data specifications by means of hierarchies of business processes, modelling elements appearing at higher levels of the hierarchies having more abstract meanings than those at lower levels. Kolb and Reichert [80] propose a framework that allows the personalisation of views of large business processes models to adapt business processes to specific user groups; taking into account parametrisable operations to hide irrelevant parts of the process. An extension of Event-driven Process Chains is proposed in [139], which is called the aggregate-EPC (aEPC). This approach combines descriptions for various related process models into an aggregated process model. A formal definition of this proposal and an add-on for the ARIS Toolset that implements this proposal are also presented.

Several approaches about reuse and abstraction involve the concept of pattern [37, 73, 150, 156]. Patterns are defined as a recurring problem that describes the core of the solution to that problem, "in such a way that you can use this solution a million times over, without ever doing it the same way twice" [5], being possible to found and adopt a solution in many contexts and situations [4]. These concepts are related because, if a pattern is identified in the definition of a business process, this pattern can be encapsulated and reused in the definition of several more business processes without having to model it in detail each time it is required.

From another point of view different to the control flow of business processes, abstraction is analysed in [109] as a mechanism to facilitate communication between resources involved in the business process execution and identifying three levels of abstraction: strategic, related to goals; tactical, related to resources; and operational, related to tasks. Finally, one proposal was found regarding to the relationship between reuse and performance, but it is not focused on the context of business processes. The proposal presented in [51] highlights the importance of reuse to improve the productivity and quality, and it introduces the importance of measuring their progress. In this publication, a categorisation of six reuse metrics and models are discussed: cost-benefit analysis, maturity assessment, amount of reuse, failure modes analysis, reusability assessment and reuse library metrics. Those are being used in industrial practice.

Although *reuse* is an accepted technique for the improvement of quality and the reduction of times in different scenarios related to business processes, as far as we know, there are not proposals focused on the implementation of reuse and the use of abstraction in the context of performance measurement and PPI definitions.

2.8 Summary

In this chapter we introduced the main concepts related to Business Process Management and Process Performance Management. We explained the Business Process Management Lifecycle and how its stages are related to process performance indicators. We focus on the description of Process Performance Indicators (PPIs) as a central element for measuring the business process performance. Several proposals used for the definition of PPIs were described, as well as the relationship between the Business Process Management Lifecycle and the PPI Management Lifecycle. The PPINOT metamodel was introduced as a mechanism for defining PPIs in an unambiguous and complete way, that has high expressiveness and facilitates traceability with the business process. Several mechanisms for modelling PPIs using PPINOT were presented: a formal definition of the metamodel, a template based notation and a graphical notation. Finally, we also presented some proposals focused on the management of business processes using reuse and abstraction as techniques for improving the Business Process Management as possible alternatives to be applied in the Process Performance Management.

Flexible Business Processes

"Flexibility is an art of creating way outs within the cul-de-sacs!"

Mehmet Murat ildan (1965),
Turkish writer and playwright

Flexibility is a characteristic of business processes related to the possibility of adapting to different scenarios, requirements or different circumstances, and which can influence all perspectives of the business process. In this chapter we focus on two main characteristics related to flexibility: variability and looseness; and in another one derived from them, decisions, as outstanding elements of business processes. Section §3.1 introduces the chapter. Section §3.2 addresses the characteristic of variability in business processes. Section §3.3 addresses the characteristic of looseness, focused on Knowledge-intensive Processes. Decisions, as a key aspect of business processes that is highly related to the flexibility of them, is addressed in Section §3.4. Finally, Section §3.5 summarises the chapter.

3.1 Introduction

In the context of business processes, flexibility is required to adapt those business processes to different changes and exceptions, to evolve business processes, to cope with business process variability and to support less structured processes which can often be characterised as knowledge intensive [137]. Predictable and repetitive business processes are suited to be fully prespecified in a process model. Traditionally, those processes have been the focus of process-aware information systems (PAIS) [137]. However, changes and new requirements of businesses have given rise to new needs in the management of business processes, requiring more flexibility.

In this dissertation, we use the taxonomy of flexibility needs proposed by Reichert and Weber in [137]. This taxonomy describes four areas that should be addressed by PAIS related to flexibility in business processes: variability, looseness, adaptation and evolution, which can affect each perspective of business processes. In Section §1.1, we briefly describe the four categories. In this research, and specifically in this chapter, we focus on two of them: variability and looseness.

Process *variability* is related to processes that need to be managed differently, deriving in different process variants [62]. Variability can be identified in several domains. The taxonomy highlights *product and service variability*, for example, in a process for evaluating an application for an insurance policy, several process variants can be generated depending on the type of insurance to be evaluated: car insurance, life insurance or vehicle insurance; variability derived from *differences in regulations*, for example, a multinational company that manufactures vehicles must comply with each country's regulations before placing the vehicle on the market: emissions of gases, power, etc.; *variability derived from different customer groups* for example, some companies use a loyalty policy to provide benefits to old customers, new customers, VIP customers, etc.; and variability related to *time*. According to the taxonomy, the second characteristic, *looseness*, is related to the *non-repeatability* because "every process instance looks slightly different", *unpredictability* since "the exact course of action is unknown and is highly situation-specific" and *emergence* of business processes as "the exact course of action only emerges during process execution when more information becomes available" [137]. These characteristics are present in the definition of KiPs, so Looseness is a concept related to Knowledge-intensive Processes.

In addition to the two characteristics of flexibility, in this chapter we also include a section dedicated to decisions related to business processes. Decisions are considered a relevant elements in business processes. Although decisions have traditionally been defined and modelled as part of business processes, the relevance they have acquired in recent years has led to the emergence of proposals for their management as an independent element of process flows.

This chapter consists of three main sections, the first one describes variability in business processes, the second addresses the characteristic of looseness from the KiPs and the third focuses on decisions. For each section an introduction to the topic is provided, modelling languages of each topic are described and related works of each topic are commented.

3.2 Business Process Variability

Business processes may exist as a collection of different variants [24, 90, 106] that share a common base structure and some strategic and business goals. When this variability is not explicitly managed, each variation in the process is modelled as an independent process of each other. This ensures the representation of all information, but depending on the amount of process variants to be defined, a long amount of models could be generated, introducing redundancy and making future adaptations difficult. The lack of control over these multiple process variants usually causes each variant takes more time to be designed, configured and modified. It also may introduce errors from the definition of variants to the evaluation of its performance [2, 138].

According to Milani et al. [106], several types of variations can be identified in a given process model or a collection of process models, which can respond to several questions. *Operational variation* focuses on *how* each variant is implemented, for example, two municipalities that solve the same problem in a different way. *Product/service variation* occurs *when* organisations produce/offer several products and the execution of the process depends on the product to be provided; for example in an insurance company the evaluation of applications for insurance policies vary depending on the type of product. *Market variation* is related to *where* the process is applied; for example, when processes need to be adapted to fit regional regulations. Another variations depend on *who* the company is dealing with; for example, when an organisation provides different sale offers depending on the customer category (VIP, new customers, etc.). Finally, *time variations* consider *when* the process is executed and is influenced by external factors, for example, if a process requires different characteristics on rainy season, on holidays, etc.

Variability has impact on the time and cost spent for the modelling and maintenance of business processes, promotes the reuse of portion of business processes, and it also has an impact on the quality of the process management, because helps management of redundancy and inconsistency [167]. Regardless of the type of variation identified, there are two ways of addressing the modelling of variability of business processes [90]. One is where each process variant is modelled separately, but it may generate redundancy and inconsistencies due to the information that process variants share. The second one is by means of a single model, from which each process variant can be derived using certain transformations; which generates less models but often more complex, thus hindering on comprehensibility. Within this second option, the management of variability can be carried out in two ways: by extension or by restriction, depending on the action performed on the unified model.

We consider that methods used for managing variability related to the control flow, data or resource of business processes, such as those mentioned above, and the benefits derived from them, such as reuse of definition and reduction of design and maintenance time, could be extended to the performance perspective in order to obtain the same benefits in the management of PPIs.

3.2.1 Variability Business Process Modelling Languages

There are many proposals to model business processes taking into account variability. Each of them focuses on a particular set of characteristics. The aim of this section is not to describe in detail all these modelling languages, but to present different ways in which variability is addressed through these languages. Therefore, we have selected a set of languages based on the proposal of La Rosa et al. [90]. This survey classifies business process modelling languages in four groups. Group 1, *Node Configuration*, in which a variation point is a node (Activities, events, and gateways) by means of which it is possible to derive a process variant; Group 2, *Element Annotation*, that are models which elements: control-flow nodes (activities, events, and gateways), sequence flows, resources and objects can be annotated; Group 3, *Activity Specialisation*, comprises business process models that relies on activity specialisation to achieve process model customisation; and finally, Group 4, *Fragment Customisations*, that groups process models based on the application of change operations to restrict or extend a customisable process model. In this section we focus on groups 1 and 4, because they contain two of the most commonly used proposals in the literature according to the number of citations listed in [90].

Configurable Integrated Event-driven Process Chains (C-iEPCs)

Configurable integrated event-driven process chain (C-iEPCs) [88, 89, 143] belongs to Group 1, *Node Configuration*. C-iEPCs are an extension of the EPC language. An iEPC is an EPC with resources and objects assigned to activities. A C-iEPC model captures the least common multiple of a family of iEPC variants. Configurable nodes are used to indicate differences between different process variants. Each configurable node can be assigned a set of customisation options, each referring to one or more process variants. Customisation of the model is made by restricting the behaviour of the C-iEPC by assigning one customisation option to each configurable node. An iEPC is derived from a C-iEPC by removing all options that are no relevant, which represents one of the original process variants given in the process family. Activities and gateways can be marked as configurable with a thicker border. Events cannot be customised. Figure §3.1 shows an example taken from [90] to illustrate a C-iEPC of postproduction process. Configurable gateways can be customised to an equal or more restrictive gateway. A configurable OR can be left as a regular OR (no restriction), or restricted to an XOR or to an AND gateway. Moreover, the number of its outgoing flows (if the gateway is a split) or the number of its incoming flows (if a join) can be restricted to any combination (e.g., two flows out of three), including being restricted to a single flow, in which case the gateway disappears.

PROVOP

PROVOP belongs to the group of approaches based on the application of change operations to restrict or extend the customisable process model, the group 4. PROVOP achieves customisation of a business process model by means of change operations applied to a set of adjustment points defined over a base process model. This base process model represents a particular domain, such as the superset of all variants or their intersection.

To configure a process variant from the base model, a set of options is proposed. Each option contains a sequence of instructions that indicates part of the business process that will be modified. These instructions are represented by change patterns: INSERT fragment, DELETE fragment, MOVE fragment, and MODIFY attribute. The first three patterns may be applied to a model fragment and the latter pattern can be used to modify the value of a process element attribute. Since adjustment points can be defined only on the control flow, in Provop it is not possible to represent variability in the resource and object perspectives.

The use of certain combinations of options can be restricted by defining option constraints, such as mutual exclusion, implication, and n-out-of-m choices. Figure §3.2 shows a based model and a set of options taken from [90]. In this example, pptions 1 and 3 are set as mutually exclusive, since Option 1 removes the adjustment point "x" required by Option 3. The rationale behind the use of these constraints is to avoid creating situations that may prevent the application of an option or that may introduce errors in the resulting variants.

3.2.2 Related Work in Variability

There are currently several proposals for the management of variability in business processes. Most of them focus on the *design and analysis* phase of the business process management lifecycle [25], wherein new BPMLs or expansion for existing ones are proposed. These languages are aimed at avoiding redundancy through the *reuse* of some parts of the business process control flow, identifying common parts of the flow and modelling a business process block only once [142]. This favours reducing duplicated information, thus decreasing design-time and maintenance-time of models [96].

For managing variability by extension, a customisable process model represents the most common behaviour or the behaviour that is shared by most process variants and then its behaviour is expanded to suit the requirements of each particular variant. Approaches such as PROVOP [61, 63] and Business Process Family Model (BPFM) [24, 105] are examples of proposals to manage variability by extension; the first one can be used with any BPML, while the second one is specific for UML Activity Diagrams. Those proposals, such as most approaches of this category allow the restriction in the behaviour of the model. For the second option, managing variability

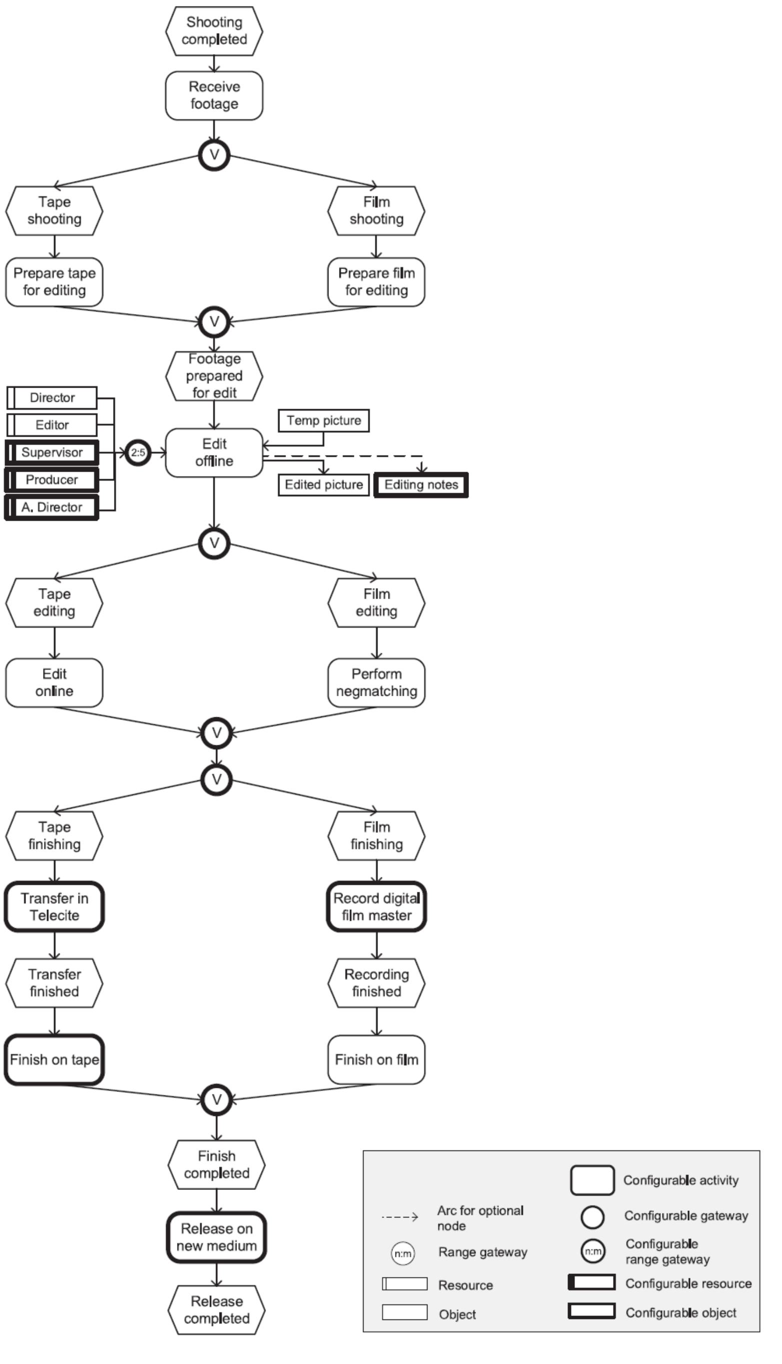


Figure 3.1: Example of a C-iEPC model taken from [90]

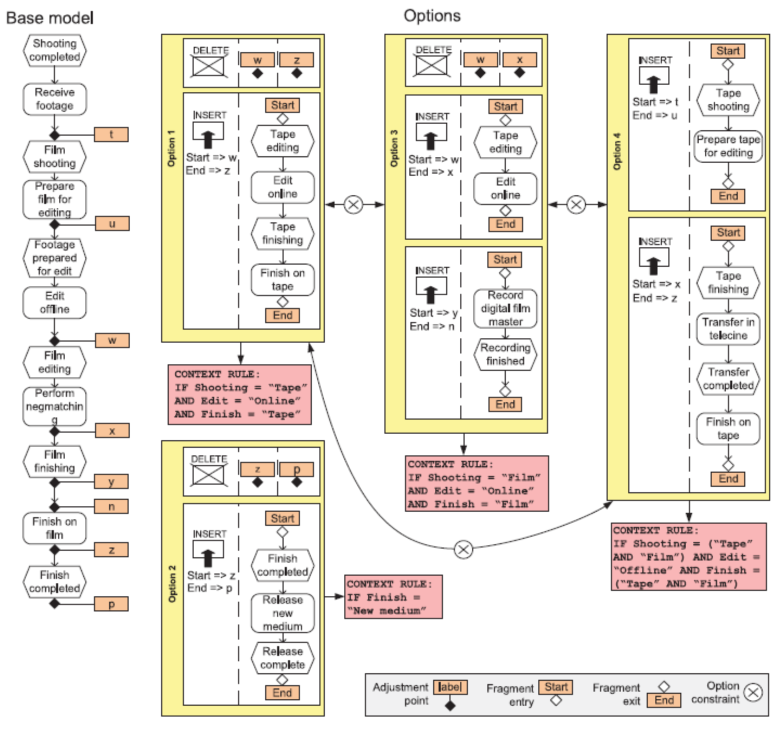

Base model
Options
Option 1
Option 2
Option 3
Option 4
DELETE
INSERT
CONTEXT RULE: IF Shooting = "Tape" AND Edit = "Online" AND Finish = "Tape"
CONTEXT RULE: IF Shooting = "Film" AND Edit = "Online" AND Finish = "Film"
CONTEXT RULE: IF Finish = "New medium"
CONTEXT RULE: IF Shooting = ("Tape" AND "Film") AND Edit = "Offline" AND Finish = ("Tape" AND "Film")
Adjustment point
Fragment entry
Fragment exit
Option constraint

Figure 3.2: Example of a PROVOP model taken from [90]

by restriction, a customisable process model contains all behaviour of all process variants and customisation is made y restricting the behaviour of the customizable process model. Customisable process models of this type are called configurable process models. Approaches such as C-EPC [143], C-iEPC [88], PESOA [147] or Feature Model Composition [1] are examples of proposals for managing variability by restriction.

Although, most related work about variability in business processes is focused on variability of control flow [61, 63, 136], there are proposals that address variability in data or resources [88, 136], but, as far as we know, there are not approaches addressing the variability of PPIs. However, in [103], some concepts about variability and indicators are treated. In particular, the variability is managed using design patterns (composite pattern), defining entities to gather goals, categories, indicators for individual, units for sets of indicators or single indicators, associated to different persons o academical units. The model proposed is based on [104], where each entity is modelled by decorator patterns, to add many features and functions dynamically. However, the authors do not deal with the traceability between PPIs and business processes and how they can vary together. They do not detail how the variability model is configured for a specific process variant and finally, the variability in indicators is described just at a high level of abstraction and it is hardly applicable in different scenarios.

3.3 Knowledge Intensive Processes

Knowledge-intensive Processes (KiPs) have been defined as a type of process that comprises sequences of activities based on intensive acquisition, sharing, storage, and (re)use of knowledge, so that the amount of value added to the organisation depends on the knowledge of the actors involved. KiPs are complex, less repeatable than conventional ones, and require a lot of creativity [75]. Based on an extensive literature review, Di Ciccio et al. [35] affirm that KiPs are processes "*whose conduct and execution are heavily dependent on knowledge workers performing various interconnected knowledge intensive decision making tasks*". Furthermore, they derived eight key characteristics typical of KiPs: knowledge-driven, collaboration-oriented, unpredictable, emergent, goal-oriented, event-driven, constraint-and rule-driven, and non-repeatable.

Additionally, Little and Deokar [94] investigate the relevance of knowledge creation in KiPs, and sustain that the expansion and use of knowledge across organisations relies on both formal and informal social processes through effective communication. Customer support, design of new products/services, marketing, management of data quality, IT governance and strategic planning are cited as examples of KiPs [98]. Develop a scientific experiment, perform medical diagnosis, and control the air traffic are other areas related to KiPs [71]. They observed that the way organisations deal with this kind of processes has changed over time, e.g. customer support processes in several organisations have evolved from highly structured to knowledge-intensive, personalised and flexible cases.

3.3.1 Business Process Modelling Languages for KiPs

Knowledge Intensive Processes Ontology (KIPO)

According to [117], an ontology is "a formal explicit description of concepts in a domain of discourse", which includes properties to describe features or attributes of the concepts and a set of restrictions to indicate how those elements are interrelated.

KIPO is a task ontology comprising the key concepts and relationships, which are relevant for understanding, describing and managing a KiP, proposed by França et al. [145]. KIPO aims to provide a common, domain-independent understanding of KiPs and, as such, it may be used as a meta-model for structuring KiP concepts. It is well-founded on UFO (Unified Foundational Ontology) [60], a foundational ontology that was developed based on a number of theories from Formal Ontology, Philosophical Logics, Philosophy of Language, Linguistics and Cognitive Psychology [59]. UFO has been used "to evaluate, re-design and integrate (meta) models of different conceptual modelling languages as well as to provide real-world semantics for their modelling constructs" [60]. It is organised in three main sections: UFO-A is the core of the ontology, focusing on endurants; UFO-B concerned with events and UFO-C dealing with social and intentional concepts [59]. Each KIPO concept is founded on one of the UFO constructs, which in turn are formally defined in terms of meta-properties (sortability, relational dependency, among others).

In a nutshell, KIPO argues that a KiP execution is driven by the agent intentions towards achieving the process objectives, and that the flow of activities (especially decision-making ones) within a KiP execution is deeply influenced by tacit elements from its stakeholders, such as Beliefs, Desires, Intentions and Perceptions [20, 135].

KIPO is structured into 5 sub-ontologies, which reflect the main perspectives that characterises a KiP. The Business Process Ontology (BPO) comprises elements encompassed within traditional business processes (such as activities, event flows, input/output data objects), which describe traditional parts of a KiP and serve as the basis from which specific KiP elements are specialised and enriched. The Collaboration Ontology (CO) depicts concepts to explain how knowledge artefacts are exchanged among process participants, and how collaboration takes place. The Decision Ontology (DO) aims to explicit the rationale of the decisions made by the process agents (i.e., the "why" and "how" decisions were made by the people involved in the process), thus allowing to track what motivated each decision and which were their outcomes. The Business Rules Ontology (BRO) provides the means to describe some parts of the KiP from a declarative perspective, since describing the rules that govern a KiP execution is especially useful for describing parts of the process which are very flexible and not subject to predefined event flows. Finally, the Knowledge Intensive Process Core Ontology (KIPCO) comprises the core concepts of a KiP, mainly Agents, Knowledge-intensive Activities and the contextual elements involved in their execution. Figure §3.3 shows the relationships between the sub-ontologies that make up KIPO. For more details about the specific components that conform each sub-ontology, see [145].

In Figure §3.4, an IT incident troubleshooting process is modelled using KIPO. This figure shows the relationship between the different components of KIPO sub-ontologies. Each element of the process modelled has a suffix that indicates the sub-ontology to which it belongs. For example, the main element is a `KIPCO::Knowledge Intensive Process` called Incident Troubleshooting, which belongs to the Knowledge Intensive Process Core Ontology. This KiP is related to `KIPCO::Knowledge Intensive Activities` such as Run Troubleshooting that belongs to the same sub-ontology; and to traditional activities such as `BPO::Activity::OpenTicket` that belongs to the Business Process Ontology. The Run Troubleshooting Activity is related to `Decisions`, `Questions` and *Alternatives* that belongs to the sub-ontology DO Decision Ontology.

Case Management Model and Notation (CMMN)

CMMN specification provides a metamodel and a graphical representation of *Cases*, as well as an interchange format for exchanging Case models among different tools. A case is described as "a proceeding that involves actions taken regarding a subject in a particular situation to achieve a desired outcome." In contrast to BPMN, used to represent predefined, fully specified, repeatable business processes, CMMN notation is useful for scenarios in which various activities need to be performed in an unpre-

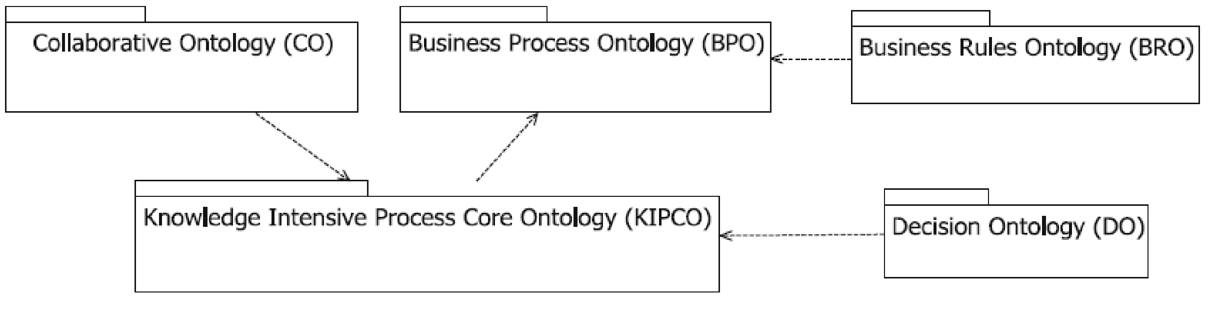

Collaborative Ontology (CO)
Business Process Ontology (BPO)
Business Rules Ontology (BRO)
Knowledge Intensive Process Core Ontology (KIPCO)
Decision Ontology (DO)

Figure 3.3: Component ontologies of KIPO. (Taken from [145])

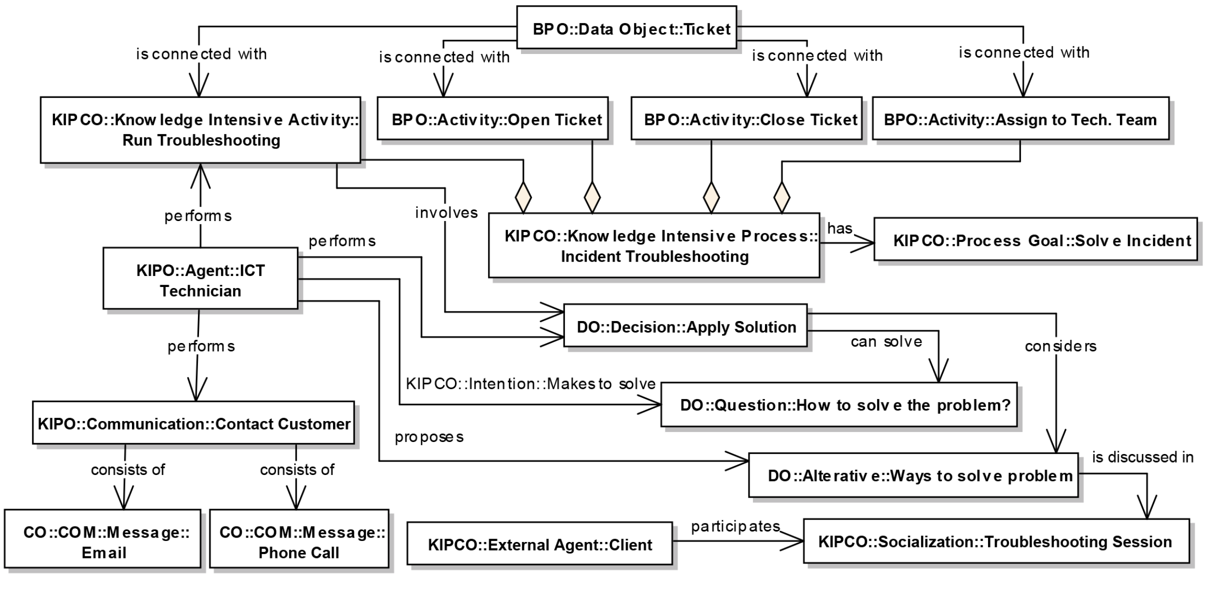
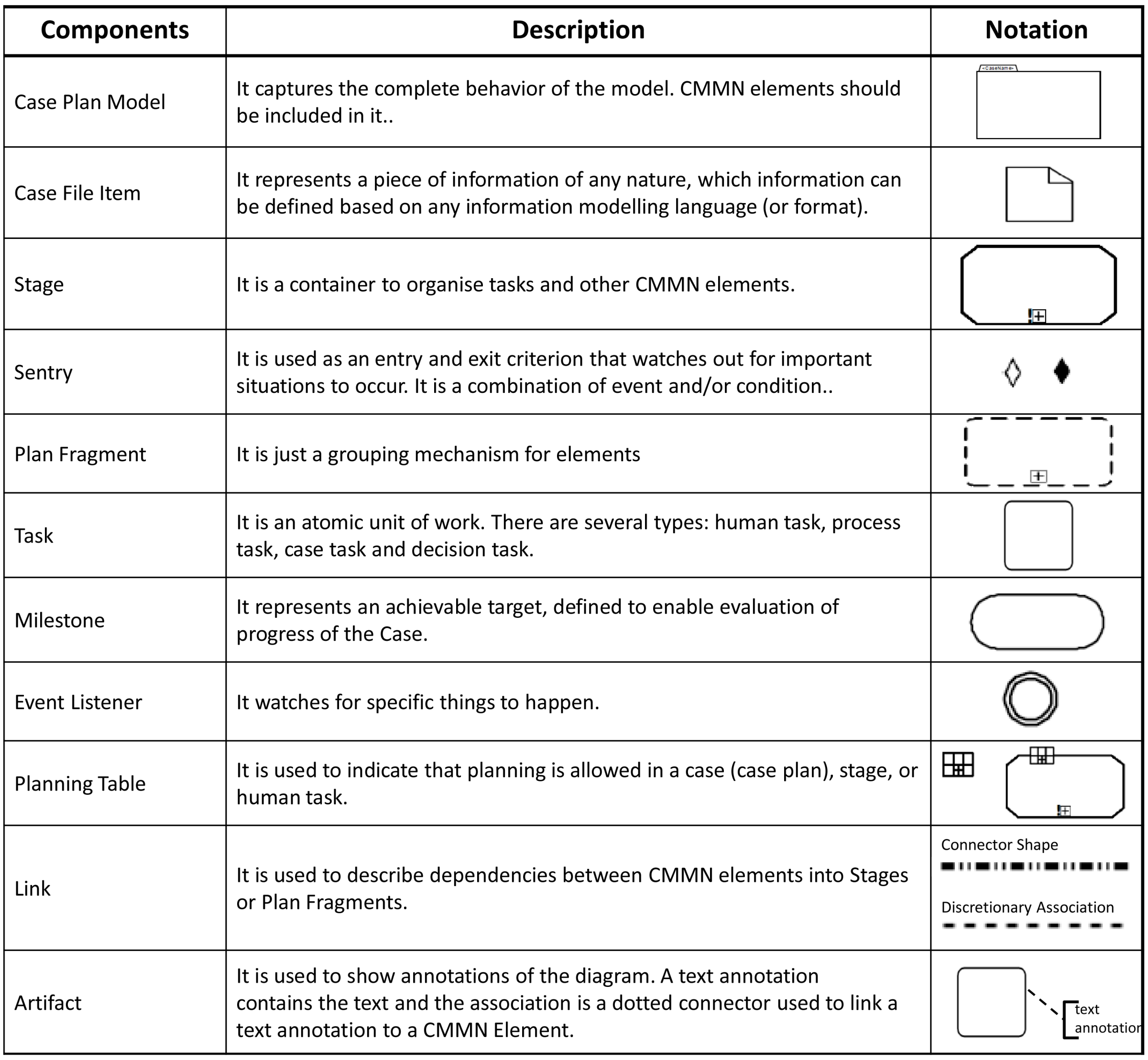

BPO::Data Object::Ticket
is connected with
is connected with
is connected with
is connected with
KIPCO::Knowledge Intensive Activity:: Run Troubleshooting
BPO::Activity::Open Ticket
BPO::Activity::Close Ticket
BPO::Activity::Assign to Tech. Team
performs
involves
KIPCO::Knowledge Intensive Process:: Incident Troubleshooting
has
KIPCO::Process Goal::Solve Incident
performs
KIPO::Agent::ICT Technician
DO::Decision::Apply Solution
performs
can solve
considers
KIPCO::Intention::Makes to solve
DO::Question::How to solve the problem?
KIPO::Communication::Contact Customer
proposes
consists of
consists of
DO::Alternative::Ways to solve problem
is discussed in
CO::COM::Message:: Email
CO::COM::Message:: Phone Call
KIPCO::External Agent::Client
participates
KIPCO::Socialization::Troubleshooting Session

Figure 3.4: Example of an Incident Troubleshooting Process modelled using KIPO



Components	Description	Notation
Case Plan Model	It captures the complete behavior of the model. CMMN elements should be included in it..	
Case File Item	It represents a piece of information of any nature, which information can be defined based on any information modelling language (or format).	
Stage	It is a container to organise tasks and other CMMN elements.	
Sentry	It is used as an entry and exit criterion that watches out for important situations to occur. It is a combination of event and/or condition..	◊ ◆
Plan Fragment	It is just a grouping mechanism for elements	
Task	It is an atomic unit of work. There are several types: human task, process task, case task and decision task.	
Milestone	It represents an achievable target, defined to enable evaluation of progress of the Case.	
Event Listener	It watches for specific things to happen.	
Planning Table	It is used to indicate that planning is allowed in a case (case plan), stage, or human task.	
Link	It is used to describe dependencies between CMMN elements into Stages or Plan Fragments.	Connector Shape Discretionary Association
Artifact	It is used to show annotations of the diagram. A text annotation contains the text and the association is a dotted connector used to link a text annotation to a CMMN Element.	text annotatio

Figure 3.5: Excerpt of the CMMN notation

dictable order.

Any individual Case may be resolved in a completely ad-hoc manner, but after several executions over time, common practices and responses can be defined for managing cases in a more rigorous and repeatable manner. Case management is often directed by a human - a Case manager or a team of Case workers - with minimal predefined encoding of the work to be performed. The care of a patient, in medical work, or the application of the law to a subject under certain situation are considered tow of the most representative scenarios where CMMN can be used.

In this section, we focused on the graphical representation of CMMN. Figure §3.5 describe its main elements. To illustrate how those elements can be used, Figure §3.6 shows a CMMN case, where the Case Write Document is modelled using most of those CMMN elements.

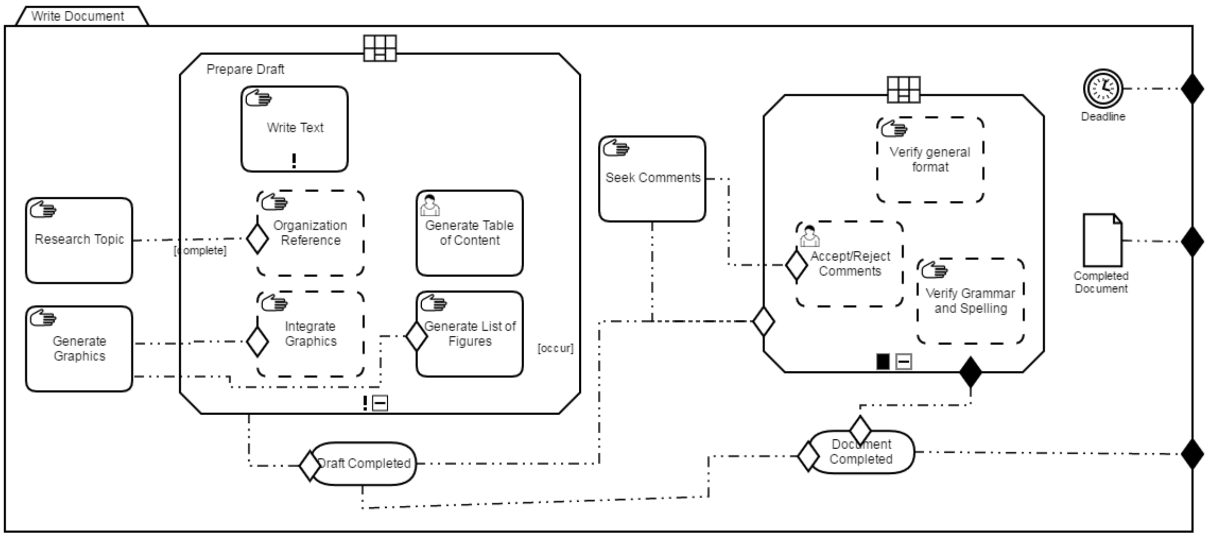


Figure 3.6: Example of a Case Write Document is modelled using CMMN notation

The *case* "Wirte document" is represented in a *Case Plan Model* as the figure of a folder. It is the container of other elements conforming elements of the case. A *Task*, depicted as a rectangle with rounded edges can be of various types. In the figure, there are two types, but both are considered human task. If the task is non-blocking it is represented with a hand in the upper left corner. If the task is blocking, it is represented with a user icon in the upper left corner. Blocking (isBlocking) is an attribute to indicate if the task is waiting until the work associated with the Task is completed (isBlocking=TRUE). If isBlocking is set to FALSE, the Task is not waiting for the work to complete and completes immediately, upon instantiation. Task can be discretionary, which are represented by having a border with a dashed line.

The figure also contains two *stages*: Prepare draft and Preliminary assessment. Both contains set of activities that can be discretionary or not, and that can be executed without following a specific order, unless the *link*-line says so. Tasks an other CMMN elements uses *decorators* to indicate specific characteristics of each element. Symbol **!** indicates the element is required and ■ indicates the element is auto-complete. For more details about decorators, see [120].

Event listeners are depicted by a double line circle shape. A *time event listener* is depicted by double line circle shape with a clock icon in the center; while *user event listener* contains a user icon inside. *Milestones* are depicted by a rectangle shape with half-rounded ends and, as in our case, can be used with *sentries* to indicate combinations of events or conditions.

In the figure, sentry of exit criterion (black diamond) is used to indicate the end of a stage or the end of the case. In our example, the case finishes because a time event occurs, or because a document (case file item) is generated (Completed document).

3.3.2 Performance Measurement of Knowledge Workers

All types of business processes, knowledge-intensive or not, need to be measured, so as to evaluate and continuously improve their performance [101]. In this section, we introduce some literature related to the performance measurement of knowledge worker and specifically on KiPs.

Measuring the productivity of a process is in general not trivial, but in the case of KiPs, there are even more challenges to be faced. These processes are typically based on human resources and how they perform their activities, and many times the result of their work is "invisible" [154]. In Section §2.6, several approaches related to the performance measurement of business processes are presented, however, none of them considers the particularities of KiPs neither in terms of other aspects of KiPs that need to be measured, such as the collaboration between process participants or the constraints and rules that drive decision making in a process execution, nor in terms of how to use these PPIs to improve process performance. In this section, we describe a few approaches in literature discuss the topics related KiPs and performance.

In [141], a classification of performance indexes is proposed to evaluate process improvement, where knowledge performance is considered as a category addressed from four measuring views: time, value (cost), quantity and quality. However, this proposal recognises the calculation of performance indexes as a challenge because there are indexes directly quantifiable (time, cost or quantity), but others related to quality require different techniques to obtain their values.

The alignment of knowledge indicators with organisations' goals is highlighted in [93]. According to this proposal, internal and external sources influence knowledge indicators. The former could be human resources (e.g. experiences, training or education level) and infrastructure (e.g. legal mechanisms or technology); and the latter may consider the general public reaction of the company, brand reputation and loyalty, and customer loyalty. The framework proposed helps the identification of knowledge asset indicators. Nevertheless, as the authors point out, how to measure those indicators has been postponed for future works.

In the research of Sturm et al. [154], requirements to measure the productivity of knowledge-intensive services were identified. They distinguish knowledge-intensive services as having a high level of complexity (high number and interrelated sub-tasks), variability (high chance of changes in activities), and uncertainty (limited availability of resources). According to the examples provided by the authors, we may infer that those types of services are normally the result of knowledge-intensive processes.

The need for a flexible evaluation system that considers multiple criteria is argued in [134]. They investigate which variables interact among each other influencing productivity. Since the production of knowledge is the key of KiP, the authors propose ways of measuring the cycle of knowledge management: creation, sharing, capturing and distribution of knowledge. They relate them to the qualification of each employee, trainings and working abilities. Besides, they also indicate autonomy of the employees, which could be related to motivation, performance readiness and lower absence. Similarly, [91] propose a metric for knowledge management performance. Therefore, they assume that knowledge circulates within the organisation, creating assets and influencing performance. They investigate the knowledge circulation process for organisational performance. The components of the index proposed are the measures for knowledge creation, accumulation, sharing, utilisation and internalisation. Both works agree that measuring performance related to processes where the main concern is knowledge is complex and involve diverse variables. [74] observed the lack of methods on how to measure effectiveness and improve KiPs. Based on results obtained through a survey, the authors argued that traditional BPM methods and techniques may not be adequate to manage and/or improve value creation in KiPs; however, there is a need to focus on managing human interaction.

In general, the above-mentioned works agree that knowledge management issues are central to determine the performance of KiPs. Some suggestions of variables have been discussed and there is a consensus that a simple system would not be able to address all of them. Proposals identified in literature do not present a conclusive approach on how to measure KiPs.

From another side, the literature on Knowledge Management advocates the relevance of considering how workers deal with knowledge to establish relationships with performance indicators [164]; [65]; [123]. Based on an empirical study, [65] argue that knowledge sharing has a strong relation with improved performance in contemporary organisational context, which is mostly based on knowledge and on how it is used within companies. They highlight that, in a knowledge-based economy, the capacity to create, transfer and adopt knowledge, rather than simply look at efficiency indicators, might regulate the long-term performance of companies.

According to [164], knowledge sharing can convert individual knowledge into organisational knowledge, and therefore improve the performance of a company. The authors investigated innovation and intellectual capital issues as critical drivers of performance in the context of knowledge sharing. [123] also tested empirically the contribution of knowledge sharing and business processes on organisational performance. For those authors, the concept of business-knowledge processes is in the scenario of activities for improving organisational performance. Their study took organisational performance as a measurement of productivity in view of the employees' knowledge contributions. They investigated three organisational operation factors: leadership support, learning and training, and communication.

Although there is a vast literature that investigates the relations among knowledge work and performance, there is not a concrete proposal for the definition of performance indicators in the context of KiPs.

3.4 Decision Management in Business Processes

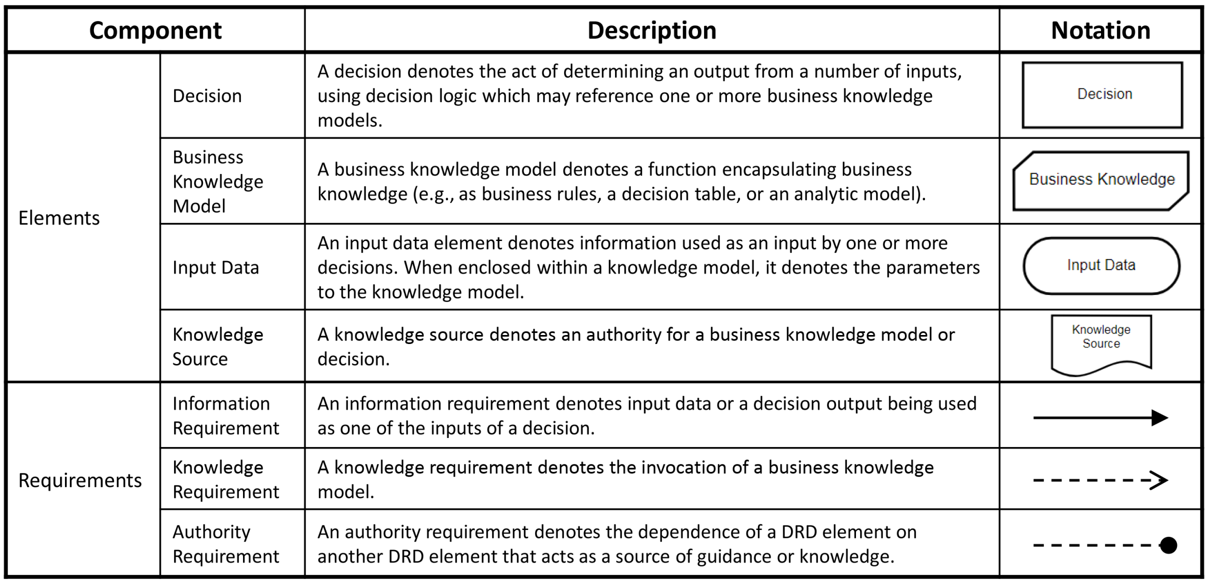
Decisions are a key aspect of every business and its processes. In previous sections, we have seen that decisions are related to both traditional business processes through their control flow and in flexible business processes, especially in Knowledge-intensive Processes.

Traditionally, decisions have been modelled either inside business process models or through decision logic using business rules or decision tables, amongst others. Recently, the Decision Model and Notation (DMN) standard [121] has been released with the aim of providing constructs to model decisions and decoupling decisions from process models. DMN can be used to model human decision-making, to identify requirements for automated decision-making and to implement those decisions.

Optimal decision making, and decision management as a more general concept, is of utmost importance for the achievement of strategic and operational goals in any organisational context. Therefore, decisions should be considered as first-class citizens that need to be modelled, measured, analysed, monitored to track their performance, and redesigned if necessary [40]. Similarly, Nura et al. [118] argue that currently, decisions are based on quantitative and qualitative proofs that can be measured by means of statistical methods for the former or using techniques like benchmarking or balance scorecard for the latter. In addition, they claim that by means of decision measurement organisations can set targets and get feedback on the progress made towards their objectives.

Regarding the analysis of a decision, several approaches agree on the importance of differentiating the *quality of the decision* that is judged by the process followed to reach the decision; and the *quality of its outcome* and the associated consequences [38, 69, 79]. According to those authors, a good decision does not guarantee a good outcome because of uncertainty presented in the decision process; and just looking at the decision outcome does not provide information about the quality of the decision.

Most scenarios found in the literature evaluate decisions on the basis of the knowledge and preferences of the decision makers, such as in [38] and [3]; and few information is taken from evidences in an objective manner, or is related to the process in which the decision takes place. Decisions are also studied in the context of business processes. However, authors have focused on the modelling of decisions and the analysis of the definition of decisions themselves, in terms of accuracy, certainty, consistency, covering and correctness [18, 72, 133]; using performance values to define decision rules [11] or providing languages for the definition of those decisions [127]; but to the best of our knowledge, no prior integrated work exists that analyses the relationship between decisions modelled in DMN and process performance and that evaluates decision performance itself based on data from event logs.



Component		Description	Notation
Elements	Decision	A decision denotes the act of determining an output from a number of inputs, using decision logic which may reference one or more business knowledge models.	Decision
	Business Knowledge Model	A business knowledge model denotes a function encapsulating business knowledge (e.g., as business rules, a decision table, or an analytic model).	Business Knowledge
	Input Data	An input data element denotes information used as an input by one or more decisions. When enclosed within a knowledge model, it denotes the parameters to the knowledge model.	Input Data
	Knowledge Source	A knowledge source denotes an authority for a business knowledge model or decision.	Knowledge Source
Requirements	Information Requirement	An information requirement denotes input data or a decision output being used as one of the inputs of a decision.	——→
	Knowledge Requirement	A knowledge requirement denotes the invocation of a business knowledge model.	- - - - - - ->
	Authority Requirement	An authority requirement denotes the dependence of a DRD element on another DRD element that acts as a source of guidance or knowledge.	- - - - - - - ●

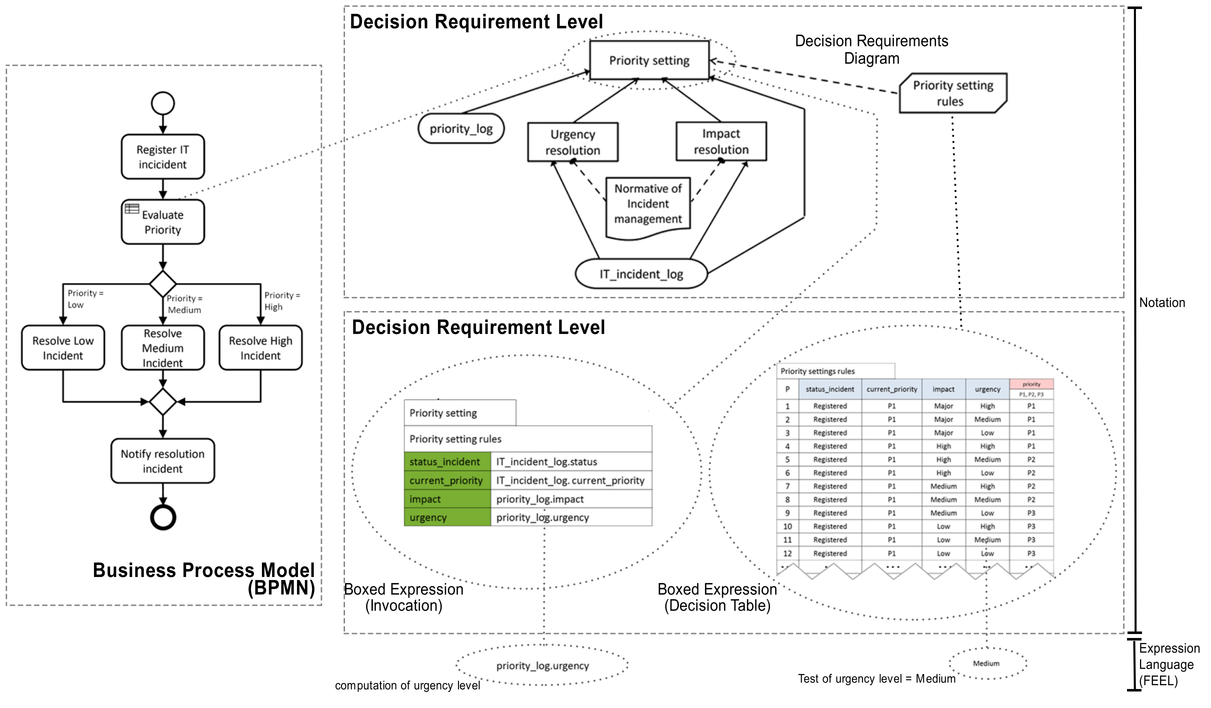
Figure 3.7: Components, elements and requirements of the DRD Notation (Taken from [121])

3.4.1 A Modelling Language: Decision Model and Notation (DMN)

Decision Model and Notation (DMN) [121] is a standard that provides constructs for describing and modelling repeatable decisions within organisations. It provides a readily understandable notation by business and IT users and ensures that decision models can be automated and interchangeable.

DMN is composed of two levels: the decision requirements level and the decision logic level. According to the DMN standard, the decision requirement level "consists of a Decision Requirements Graph (DRG) depicted in one or more Decision Requirements Diagrams (DRDs)". A DRG shows the most important elements and dependencies involved in a domain of decision-making, while DRD is considered a partial or filtered view of a DRG. Both the elements of a DRD and the dependencies between them are presented in the DMN notation that is briefly described in Figure §3.7.

For the second level, the decision logic level, the Friendly Enough Expression Language (FEEL) is provided for defining and assembling decision tables, calculations, if/then/else logic, etc. In addition, a notation for decision logic, called boxed expressions", is provided to graphically represent those expressions and to show their relationship with elements of a DRD. Figure §3.8 shows a linkage between the two DMN levels and a business process model. The business process is represented using BPMN, but DMN is not dependent on BPMN. DMN can be related to other standards and its two levels may be used independently or in conjunction to model a domain of decision-making without any reference to business processes. This example includes the business process model to register and solve an IT incident in a company. As we can see, this business process has an activity where a decision should be made.


Decision Requirement Level
Decision Requirements Diagram
Priority setting
Priority setting rules
priority_log
Urgency resolution
Impact resolution
Normative of Incident management
IT_incident_log
Register IT incicident
Evaluate Priority
Priority = Low
Priority = Medium
Priority = High
Resolve Low Incident
Resolve Medium Incident
Resolve High Incident
Notify resolution incident
Business Process Model (BPMN)
Decision Requirement Level
Boxed Expression (Invocation)
Boxed Expression (Decision Table)
priority_log.urgency
computation of urgency level
Medium
Test of urgency level = Medium
Notation
Expression Language (FEEL)

Priority setting

Priority setting rules	
status_incident	IT_incident_log.status
current_priority	IT_incident_log. current_priority
impact	priority_log.impact
urgency	priority_log.urgency

Priority settings rules

P	status_incident	current_priority	impact	urgency	priority P1, P2, P3
1	Registered	P1	Major	High	P1
2	Registered	P1	Major	Medium	P1
3	Registered	P1	Major	Low	P1
4	Registered	P1	High	High	P1
5	Registered	P1	High	Medium	P2
6	Registered	P1	High	Low	P2
7	Registered	P1	Medium	High	P2
8	Registered	P1	Medium	Medium	P2
9	Registered	P1	Medium	Low	P3
10	Registered	P1	Low	High	P3
11	Registered	P1	Low	Medium	P3
12	Registered	P1	Low	Low	P3
...	...	...	...	...	...

Figure 3.8: DMN Constructs, DMN Levels and its relationship with the business process

In this example, it is possible to see that the relationship between the business process model and the Decision Requirement Level is made by means of an activity of the business process where a decision is required. In this level, the main element of the DRD is a decision (Priority Setting) that requires information from other two decisions (Urgency resolution and Impact resolution) and from two input data elements (priority_log and IT_incident_log). The *Priority Setting* decision is related to business knowledge model elements, of which "Priority Setting rules" is the link between the decision requirement level and the decision logic logic level. The use of the FEEL language is also shown in this figure.

The DMN standard makes reference to two possibilities to define a decision. One is used for "the act of choosing among multiple possible options" and the other one "may denote the option that is chosen". However, its documentation specifies that DMN adopts the decision concept as "the act of determining an output value (the chosen option), from a number of input values, using logic defining how the output is determined from the inputs. This decision logic may include one or more business knowledge models which encapsulate business know-how in the form of business rules, analytic models, or other formalisms".

Finally, the relationship between the DMN components is supported by the DMN metamodel also described in the DMN standard. Figure §3.9 shows an excerpt of this metamodel, whose focus is the `Decision` class and its relationships with other DMN components. Although DMN elements such as decision tables are not directly iden-

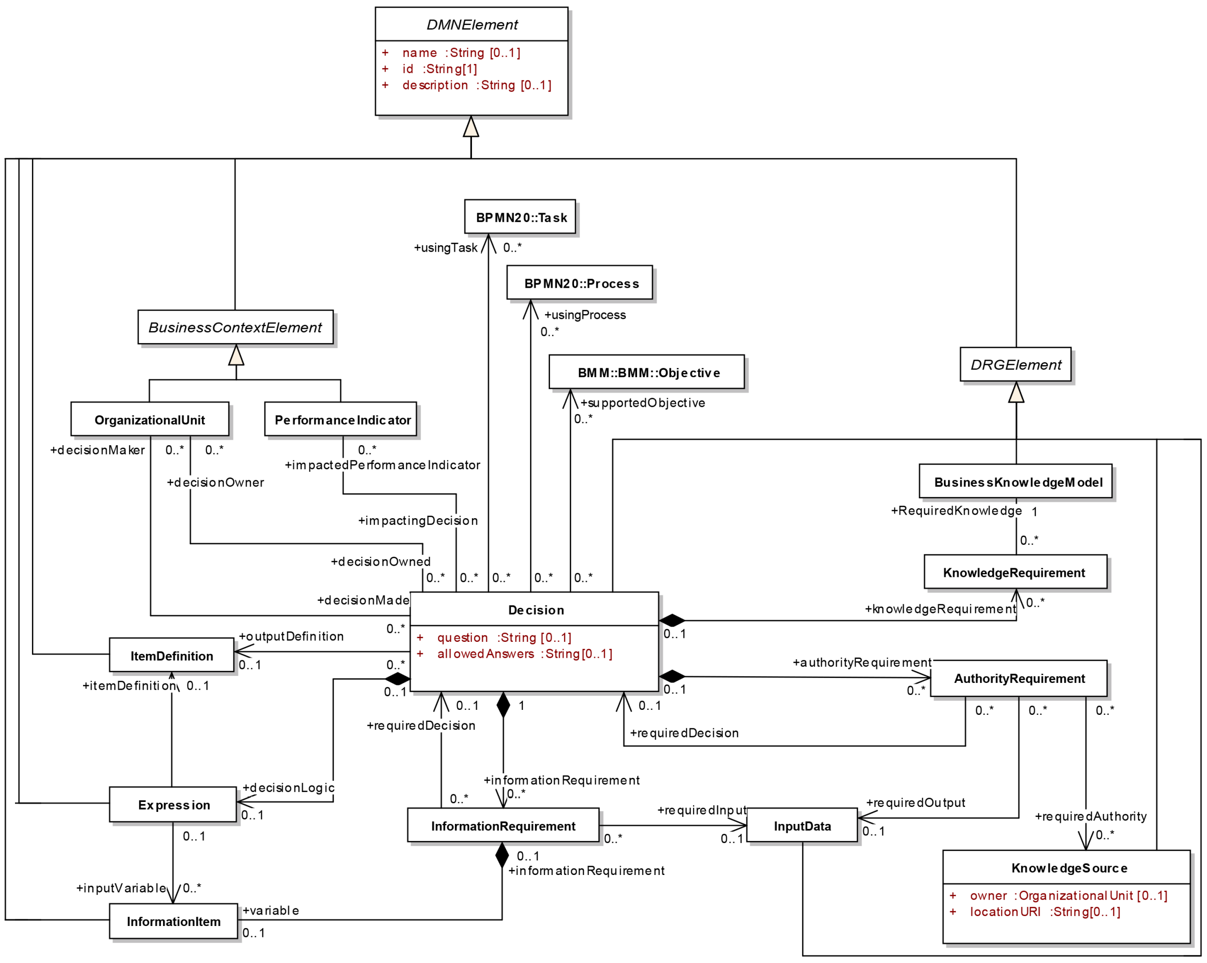


Figure 3.9: Decision Metamodel provided by DMN to show relationship between their elements. (Taken from [121])

tified in the DMN metamodel, the components of these decision tables are included in it. For example, in the diagram of the DMN metamodel, we can identify inputs of the decision table by means of the `InputData` class and its outcomes are represented by means of the `ItemDefinition` class. In the context of this research it is important to highlight the existence of another relationship, `impactedPerformanceIndicator`, which connects the Decision class with the PerformanceIndicator class. The latter class is considered a `BusinessContextElement`, which in turn is a `DMNElement`, but the `PerformanceIndicator` class is not represented in the DMN graphical notation.

3.4.2 Related Work in Decisions and Performance

Relations between business processes and decisions, and specifically decisions that can be modelled in DMN, have been addressed in different approaches. For example, [160] proposes the integration of processes and decision modelling using BPMN and DMN, [10] derives decision models from business processes, [76] relates processes and decisions by means of a set of integration scenarios, or [9] presents frameworks for adjusting decision models dynamically according to the business process environment and ensure SLA compliance, to name a few. However, process performance is not considered in the context of these relationships.

Other proposals focus on the quality of the logic expressed in decision tables. To this end, measures such as certainty, consistency and covering are proposed to evaluate a set of decision-rules extracted from a complete [132], incomplete [131] or an ordered decision table [133]. In the same vein, [72] proposes an algorithm for measuring rule set consistency evaluating similarity between different rule sets; and [18] proposes algorithms for correctness checking tasks over DMN tables. However, they do not evaluate each decision instance, but the decision model expressed as decision tables.

More related to our proposal are [11, 54], which are related to the impact of decisions in process performance. Specifically, [54] derives decision criteria formulated as decision rules based on experience gained through past process executions, although they do not consider the specifics of the DMN standard. Concerning [11], the authors propose a formal framework to derive decision models from event logs using DMN and BPMN and taking into account predictions of PPIs. However, they are concerned with obtaining decision models instead of helping to understand the consequences of each decision. Regarding the performance measurement of decisions, [3] addresses the quality of customer decisions using measures mostly based on preferences of decision makers and not on objective data taken from the process. Finally, the use of performance indicators in the definition of decisions is dealt with in [127], which proposes a query language to extract information from process or task instances that allows definitions of measures in boxed invocation.

3.5 Summary

In this chapter, we addressed flexibility in business processes since three different points of view. First, we addressed flexibility since the characteristic of variability in business processes. Second, we focused in the characteristic of looseness in business processes, specially addressing Knowledge-intensive Processes. And finally, we also addressed decisions, as key aspects in business processes and as elements that day by day have become more and more relevant in the context of the business processes, to the point of having their own notation of representation. For each characteristic addressed, we introduced main concepts related to them; described certain modelling languages or notations for the representation of each characteristic in a specific scenario; and finally, we also presented some approaches considered related work to each of the points of view addressed.

Part III

OUR PROPOSAL

Reuse and Abstraction in PPI Definitions

"Good programmers know what to write. Great ones know what to rewrite (and reuse)."

Eric S. Raymond (1957),
American software developer, author of The Cathedral and the Bazaar

Organisations often define a large number of business processes to describe their day-to-day activities, which can generate large repositories of information where in many cases there is redundancy between these processes. Definition and modelling of these business processes may require large amounts of time and effort and its maintenance can be susceptible to errors given the large amount of information involved. This redundancy is not only present in the business processes control flow, but also in its performance perspective. In this chapter, we address redundancy by means of the reuse of PPI definitions using abstraction concepts. Introduction to this chapter is presented in Section §4.1. Section §4.2 describes an scenario based on the SCOR model in which the problem is motivated. Section §4.3 proposes an alternative to improve PPI definitions by means of the reuse of some PPIs or parts of them. Section §4.4 shows how reuse can be implemented in PPI definition using PPINOT. Templates and graphical notations of PPINOT are extended in Section §4.5 and Section §4.6, with the aim of illustrating how to model PPIs taking into account abstraction concepts. A discussion about the contributions of this chapter is presented in Section §4.7. Finally, Section §4.8 summarises the chapter.

4.1 Introduction

Organisations typically define a large number of business processes [36]. Considering that each business process can be related to several PPIs, big repositories of information can be generated by organisations after definition and modelling of all their business processes and PPIs. In this respect, the graph shown in Figure §4.1 represents the relationship between business processes and the set of performance measures defined for each one. This is a real example constructed from the SCOR model [6], a process reference model for the supply chain management that proposes and defines measures (metrics) and business processes related to the supply chain (see Apendix §A). In our example, a total of 232 business processes are considered, from which only 50 processes do not have measures defined and the remaining processes are related at least to one measure: 35 processes have 1 measure defined, 27 processes have 2 measures, 36 processes have 3 measures, and so on up to 3 processes with 27 measures defined on each one; describing a total of 969 measures.

Information gathered in those repositories might be redundant. The modelling of the set of similar business processes requires a lot of time to design, model and manage them [99] and also requires tools and techniques that guarantee a correct state of the process after its management and modifications, because in some cases, changes should be applied over more than one version of the business process. Measures and PPIs can also generate redundant information, because they can be used for more than one business process. In our example based on SCOR, even 969 measures are related to all SCOR processes, it only defines 282 different measures. It means that almost the 71% of the definitions of measures represent redundant information because they are applied to more than one process. From the 282 measures, the 58.51% of the measures (165) are defined for more than one process and only the 41.49% of the measures (117) are defined once. From the 165 repeated measures, 11 measures are defined twice; 83 measures, 3 times; 14 measures, 4 times; 10 measures, 5 times, etc. The complete detail of redundancy of measures in SCOR is shown in Figure §4.2. In addition to this redundancy of information, many of these SCOR measures follow common patterns, such as

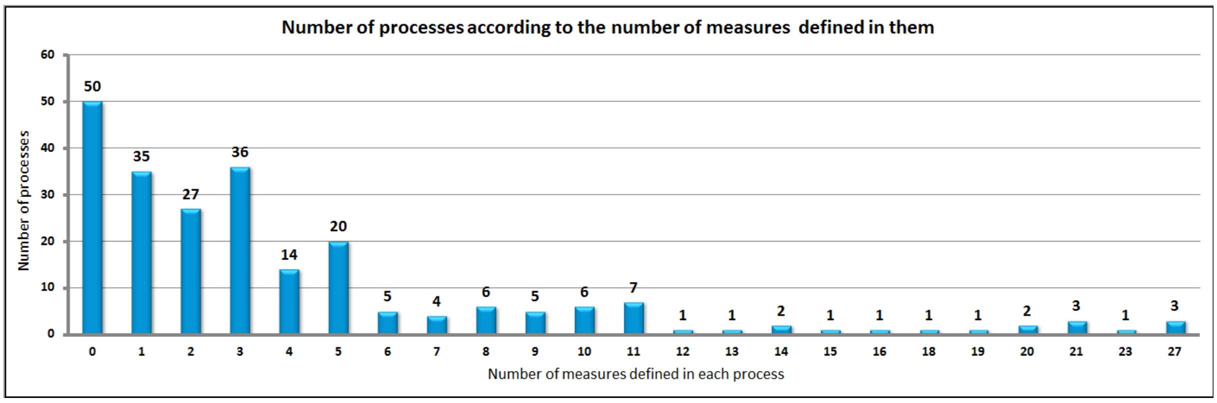


Figure 4.1: Number of processes classified according to the number of measures.

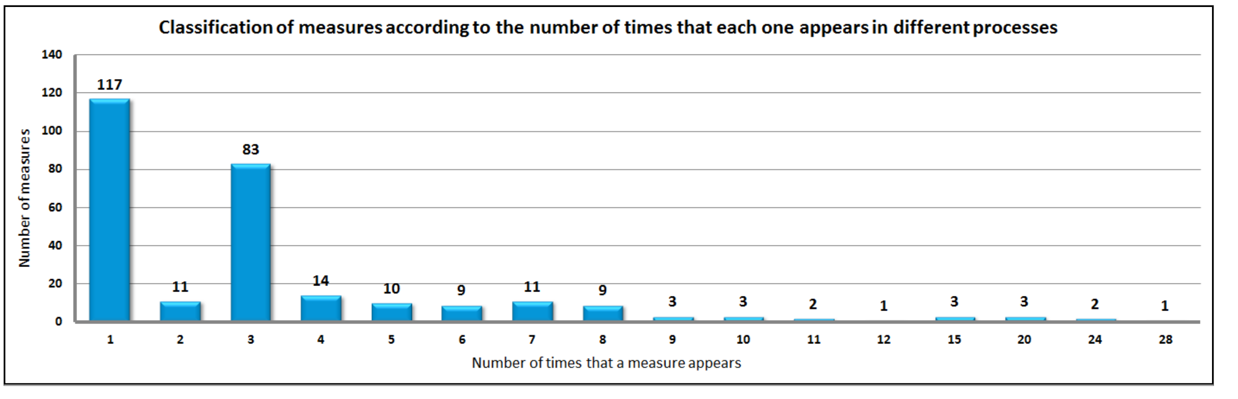


Figure 4.2: Classification of measures according to the number of times that each one appears in different processes.

formulas, percentages, etc., that could be reused.

As far as we know, there are several proposals for managing redundancy in business process models, but there are not proposals that apply this technique in the context of process performance indicators. This lack of proposals requires that, given a set of PPIs, each of them must be defined and modeled individually and completely for each of the business processes in which the PPI is defined. Individual and repetitive PPI modelling can be time-consuming. In addition, modifications to these definitions can lead to erroneous models, if for example, a PPI that is repeated for 5 processes is only updated in 4 of them, thus generating inconsistent models.

Due to the lack of alternatives related to the management of redundant definitions of PPIs, we base our proposal in a technique previously applied in the context of business processes: the *Reuse*, which allows us to design a business process model by using existing process models [99]. For proposals such as [4], the reuse of business process has a beneficial impact on the quality, efficiency and effectiveness of business process modelling; seeks to reduce modelling time by avoiding modelling the same business process or part of it multiple times [77, 99]. In this sense, *Abstraction* is one of the most used techniques to materialise the reuse and is considered an essential feature in any reuse technique [86]. In business processes modelling, abstraction helps reduce complexity in models presented, facilitate readable processes, and allows the representation of business processes by means of high levels views, showing aggregated activities and hiding irrelevant details to particular users [39, 148, 149]. Although there are different ways to implement abstraction [27, 84], the *aggregation* is the alternative that better describe most examples found in literatura. In this paper, we adhere to the concept aggregation presented in [27], where "a set of elements is grouped hierarchically under a higher-level element, perhaps of a different type, which serves as an aggregate".

In this chapter, we aim at improving the measurement of performance in business processes by means of the reuse of PPI definitions or part of them, with the aim of reducing drawbacks derived from the redundancy in PPI definitions. To do this, we describe a scenario based on the SCOR mode in order to determine different reuse cases existing in PPI definitions. We also propose the formalisation of these reuse cases for the definition and management of PPIs by means of an extension of the *PPINOT metamodel* [31]. Finally, notations of the PPINOT metamodel are extended to illustrate how the reuse can be applied and modelled using PPINOT taking into account reuse and abstraction concepts.

4.2 Problem Illustration

The problem illustration presented in this section is based on the SCOR model and other PPI projects in which we have been involved. As we describe in Apendix §A, SCOR is a process reference model for the supply chain management, which describes business activities related to all needed phases to the supply customers demand and integrates concepts of business re-engineering, benchmarking, performance measurement and provides a standard format to facilitate communication [70]. It also enables users to address, improve and communicate supply change management practices within and between all interested parties in the organisation [128]. In Appendix §A, the SCOR model is described in detail.

The SCOR model has 282 different measures: 7 of Level-1, 32 of Level-2 and 243 of Level-3, most of them are associated to more than one process. From the 282 measures, the 58.51% of the measures (165) are defined for more than one process and only the 41.49% of the measures (117) are defined once. Figure §4.2 shows this information in more detail. There are 117 measures defined once, 11 defined twice, 83 defined 3 times, etc. Figure §4.1 complements this information from the point of view of processes. In this figure, SCOR processes are classified according to the number of measures defined for each of them. For example, there are 50 processes with no measure defined, 31 processes with only one measure, 27 process with two measures, and so on up to 3 processes with 27 measures defined. Suppose we would like to model each process with all measures defined for it, 969 measures should be modelled: 23 measures defined over Level-1 processes, 445 over Level-2 processes and 501 measures defined over Level-3 processes. The modelling and maintenance of all this measures could be a long and laborious task, being also error prone due to the necessity of repeating each measure definition more than once for different processes. When a measure definition changes, all related definitions of this measure must change, and it is not fully possible to ensure the integrity for all those measures if changes are applied manually.

In the classification of processes and measures of SCOR, presented in Figures §4.1 and §4.2, we are considering only one type or reusability: when a measure is defined for more than one process, it means that a measure is implemented and used in the same way for all the business processes where is defined. For example, the SCOR measure *Pack Product Cycle Time (RS.3.95)* calculates the average time associated with packing a product for shipment. This measure is associated with three Level-3 pro-

Table 4.1: General patterns identified in SCOR

Pattern	Description	*N*° Measure
SUM	Sum of values of the same type, such as time or cost.	48
AVG_Time	Average calculated using time values.	114
PCT	Percentage calculated with a set of values.	76
CNT	Counting of conditions	9
FUN_5	A predefined SCOR function is applied. 5 point rolling average is calculated with four previous quarters values and one projection.	8
NoPattern*	A particular formula is calculated over a set of values.	27
Total of measures		***282***

* This is not a pattern. This is a set of measures that does not correspond to another category.

cesses *Deliver* and in each process the measure is defined in the same way. For this example, the PPI definition and specifically the measure that defines it, only changes when the source of data is defined.

However, there is another type of reusability in SCOR measures, which is most related to the use of PPI patterns. To calculate a PPI value, a set of data inputs are required to indicate information to be used in its calculation. In some cases, a PPI may be defined as a generic pattern that does not change the formula or function used for the calculation of the PPI value, but its number of data inputs required may change. For example, suppose we have a set of PPIs whose purpose is to calculate the *average execution time* of a set of business process tasks. For one process, a PPI that uses this measure may require the execution time of three task, but for another process four, five or more task may be required. In this case, the measure definition is the same, but the number of inputs is different. Table §4.1 shows a first classification of SCOR measures according to its purpose and expected value. From the total of 282 measures we propose 6 categories, where the first 5 values of the first column are possible patterns of measures. The last row (*NoPattern*) is a special category because it does not represent a pattern. It brings together all measures which could not be included in other category.

Finally, although it is not directly related to SCOR's types of measures, we also have identified another type of reuse. Most of PPIs are calculated taking values exclusively from business process elements (e.g. activities, data objects, etc.), but in some cases may be useful to provide data from external sources such as data provided by users. This can be seen as a parametrisation of PPIs. For example, the *Andalusian Health Service* defines a set of PPIs for measuring the percentage of incidents resolved in a period of time. Depending on a priority value established, the tolerance of the period of time changes. The higher the priority level, the shorter the expected resolution time and vice versa. The priority and the resolution time expected are external values that should be provided as a parameter of the PPI.

Considering the scenario described above we formulated a research question to help guide research on the relationship between the reuse and business process performance.

R-RQ *How can the reuse be implemented in definitions of process performance indicators?*

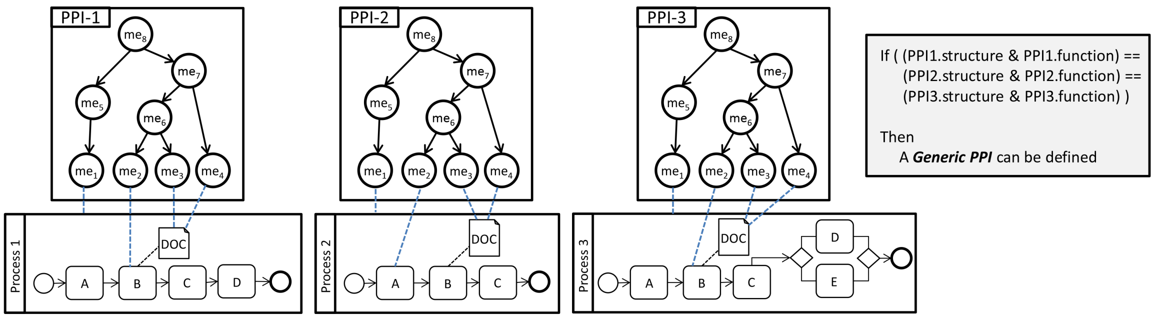
4.3 Abstraction in PPI Definitions

In this section, we relate the cases of reuse described in the previous section to the definition of PPIs as mechanisms to measure the performance of business processes. We consider that abstraction is a valid and appropriate method for dealing with the cases of reuse raised, because abstraction is conceived as an operation that help produce a simpler model but retains properties of interest from a original model [27].

Several criteria are proposed to classify abstraction [27, 84] depending on the characteristics that should be represented. In this proposal, we focus on two concepts: *Hierarchies* and *Aggregation*. *Hierarchies* allow the representation of a model by means of different views, hiding non-relevant details of a definition; while *aggregation* relates a model with its set of components by means of a *is-part-of* relationship. From the point of view of reuse and modelling of business processes, hierarchies are closely related to the concept of *Abstraction Level* [169]. An abstraction level is made of a part of the business process model that represents a function frequently used. The advantage of abstraction levels is that does not require complex operations to be adapted to requirements of a particular business processes and can be reused through a instantiation of the process.

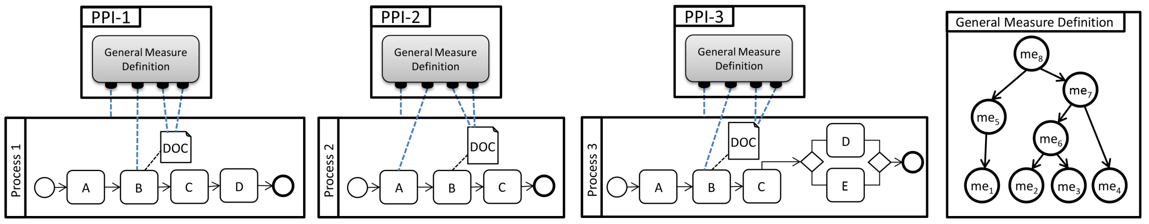
In this way, we propose the concept of abstraction level can be extended to performance models. We seek to represent a PPI definition as a set of abstraction levels, where non-relevant definition details can be hidden by means of new elements that groups them and represent the meaning of the original elements. Each general definition is linked with a detailed definition that contains all the information hidden in the upper level. In Figure 4.3(a) three different business processes models are presented. Each of them has one PPI defined. Each PPI contains a measure definition specifying how the PPI should be calculated. Suppose those PPIs are equivalents, which means that those PPIs have the same structure, receive the same input values and all of them are calculated using the same formula. Those PPIs can contain reusable measure definitions that can be use in several PPI definitions. In Figure 4.3(b), we represent a reusable PPI definition that only defines the general purpose of the PPI and the set of inputs required, and details of the measure are detailed described in a separately view (or other level).

We call *Composite Measures (CM)* the general elements that substitute details (measures) of a PPI definition. A *CM* may be one of two types: *Expanded-CM* that can be seen as a container that represents a set of measures interconnected that can be reused in other PPI definitions. The importance CM is to provide the possibility to define a set of measures only once, but can be used many times by means of a *Collapsed-CM*,



PPI-1
PPI-2
PPI-3
me8
me7
me5
me6
me1
me2
me3
me4
DOC
Process 1
Process 2
Process 3
A
B
C
D
E
If ((PPI1.structure & PPI1.function) ==
(PPI2.structure & PPI2.function) ==
(PPI3.structure & PPI3.function))
Then
A Generic PPI can be defined

(a) Example of a PPI definition in three business process models



(b) Example of a PPI definition using abstraction.

Figure 4.3: Example of PPI definition represented in original way and using abstraction.

which is an instantiation of an Expanded-CM. Both are directly related. A Collapsed-CM cannot exists without the previous definition of an Expanded-CM. They should be linked by an identifier and should have the same number and types of inputs. In a CM, each input required is called *Connector*, because it allows the connection between two elements: one is a composite measure that calculates a value using inputs and a set of measures; the other element can be a measure required to calculate a value or an external value. The measure obtains information from the business process and the external value represents a value not obtained from the business process, it can be provided by the user.

To provide more flexibility and to facilitate the reuse of PPI definitions, we propose to include *external values* as *parameters* of CMs. It allows us to define a PPI capable of receiving data from sources different than the business process elements, for example using data provided directly by a user. To establish the connection between composite measures and external values it is necessary to use *Parameter connectors* as a particular type of *connector* of a CM. Parameters, in this context, can be numerical or textual values, conditions, functions, etc., almost anything that determines how a measure should be calculated, but it cannot be another measuring element such as a measure or a business process element. For example, suppose that there is a PPI to calculate the percentage of delay in the resolution time of an incident and that there are different types of priority (low, medium and urgent) for the incidents. For each priority, the resolution time varies. An example of external value can be the maximum number of hours allowed to consider that an incident is performed on time: 24 hours for a low priority incident, 8 hours for a medium priority incident and 2 hours for a high priority incident. In this example, 24, 8 and 2 hours are parameters of the measure definition and the PPI.

In our analysis of SCOR measures, we identified some measures that apply a predefined function over a set of data values, regardless of the number of data inputs provided; the sum or the average of execution time or the sum of costs of a process, for instance. In a traditional PPI definition it is necessary to determine the number of inputs to be used in the calculation of the measure. To cope with this issue, we propose to include a new concept in PPI definitions, the *List* concept. A List returns a single value as a result of a function applied over a set of elements (measures or external values) without the need to specify the number of inputs that it contains. An important characteristic of a List is that all its elements should have the same type, in other words, it is possible to define a list of numbers, a list of data, a list of boolean values, etc., but a list cannot contain a combination of those. For example, we could define a measure that calculates the average execution time of certain activities. A measure in one PPI could be defined using time value information from 3 activities and a measure in another PPI could be defined using time values from 7 activities. The function applied to both measures does not change, an average is calculated, but the amount of values used changes.

4.4 Abstraction in PPINOT

Although the proposal to use abstraction concepts to encourage the reuse of PPIs can be applied to different proposals for the definition of indicators regardless of the business process modelling language used, its materialisation must be personalised to each proposal of PPI definitions. In this chapter, as in the next chapters related to our contributions, we will base our proposal on the PPINOT metamodel. In this section, we describe how the PPINOT metamodel should be extended to support abstraction concepts with the aim of reuse PPI definitions or part of them. We based this extension in the metamodel presented in Section §2.6.

The UML diagram shown in Figure §4.4 represents the core of the PPINOT metamodel and its extension considering abstraction concepts to reuse PPI definitions. The original PPINOT classes are represented by white boxes. White boxes with the name underlined represent business process elements with which the PPINOT elements are related. New classes that extend the metamodel are shown as gray boxes.

The extension of the PPINOT metamodel can be summarised in two points: the first one is the incorporation of a new type of `MeasureDefinition` called `CompositeMeasure`, which can be of two types `CollapsedMeasure` or `ExpandedMeasure`; the second point is the incorporation of a new `DerivedMeasure` called `ListMeasure`. As we explained in the previous section, the `CollapsedMeasure` can be seen as a container of measures (`BaseMeasures` and/or `AggregatedMeasures`); while the `ExpandedMeasure`

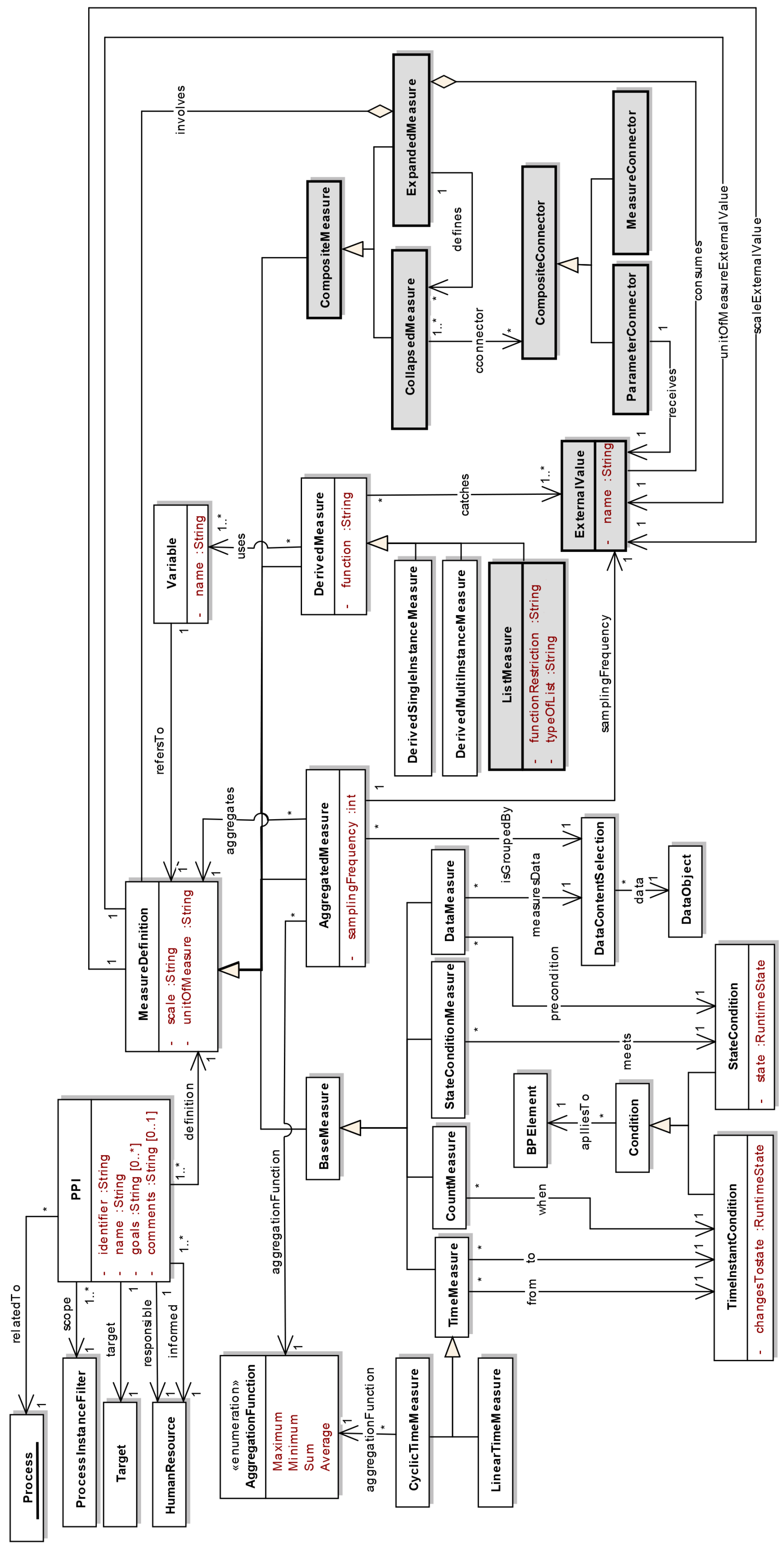


Figure 4.4: PPINOT Metamodel extended to support reuse in PPI definitions.

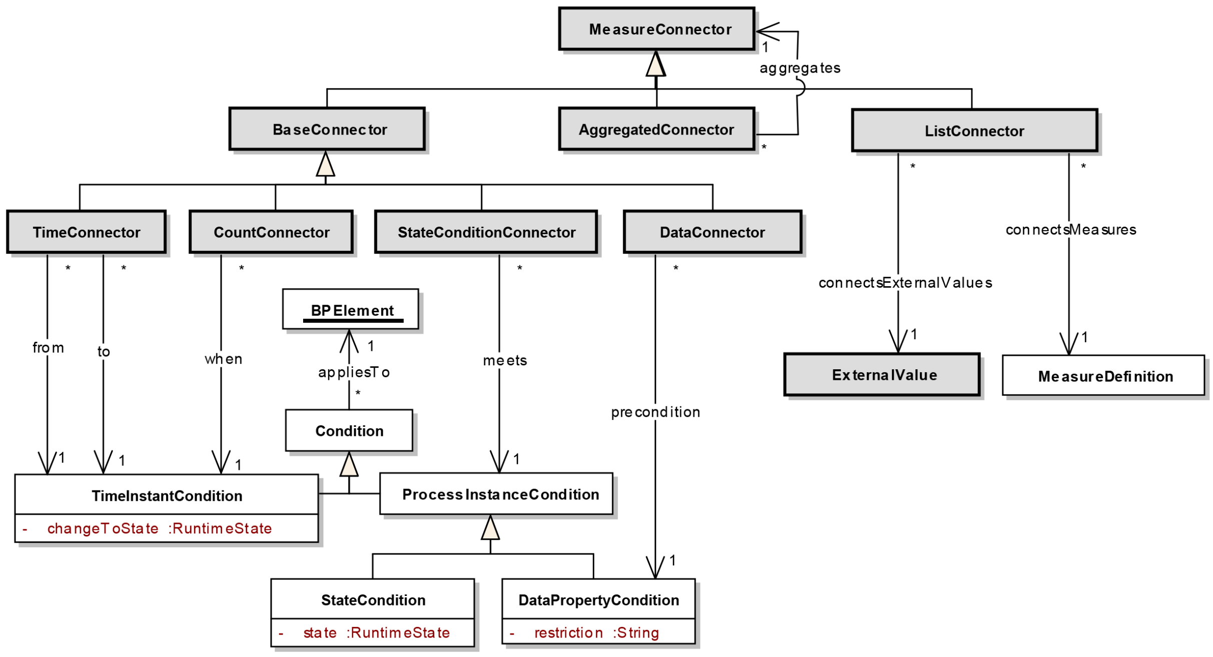


Figure 4.5: Connectors used in the PPINOT metamodel extension for abstraction.

contains the detail of all measures required in a PPI definition. In an ExpandedMeasure it is possible to define a pattern to be reused as a structure of measures connected to each other, identifying which elements should be provided to calculate the measure value but that are not connected with business process elements. It is possible to say that an ExpandedMeasure *defines* one or more CollapsedMeasure. This relationship makes sense when we want to check the relationship between a CollapsedMeasure and an ExpandedMeasure and we want to verify that the instantiation of the CollapsedMeasure is correctly connected with elements that should provide information for the calculation of the measure and the PPI. In a business process model that includes PPI definitions (PPI models), the ExpandedMeasure is not included. CollapsedMeasure appears as a representation of the set of measures defined in an ExpandedMeasure.

A CollapsedMeasure requires at least one CompositeConnector, which can be of two types: ParameterConnector and MeasureConnector. The first type, ParameterConnector, links a CollapsedMeasure with an ExternalValue. An ExternalValue represents information that is not taken from the business process and that is provided by final users. On the other hand, a MeasureConnector links a CollapsedMeasure with a BPElement or a MeasureDefinition by means of a Condition. As with the measures, MeasureConnectors can also be of different types and are related to a particular type of condition. Figure §4.5 graphically shows relationships between the different types of connectors incorporated in the PPINOT metamodel for the definition of CompositeMeasures and the Condition, BPElements, MeasureDefinitions or ExternalValues.

`TimeConnector` and `CountConnector` are related to a `TimeInstantCondition` by means of the the links *from, to* and *when*. `StateConditionConnector` is linked to a `ProcessInstanceCondition` by means of *meets*; and a `DataConnector` uses a *precondition* to specify a `DataPropertyCondition`. The relationships between `BaseConnectors` and `Conditions` are similar to relationships between `BaseMeasures` and `Conditions`, because they represent the same information, but `CompositeConnectors` are used in a `CompositeMeasure` to incorporate abstraction concepts. `AggregatedConnector` aggregates a `BaseConnector` or a `ListConnector`.

Notice that there is not a `CompositeConnector` for `DerivedMeasures`, just for the `ListConnector`. This is because a `DerivedMeasure` can involve different types measures and/or external values, so those elements should be specified one by one as measures or external values. The `ListMeasure`, on the other hand, does not need such a specification of its elements one by one, because all its members must be of the same type and the function of the measure applies to all of them. A `ListMeasure` class is included as a particular type of a `DerivedMeasure`. A `ListMeasure` can be connected with `BaseMeasures` or it can receive *ExternalValues* as members of its list of elements. As a `DerivedMeasure`, it is possible to define a *function* in a `ListMeasure` to specify a formula that affects all elements of the list. Unlike `DerivedMeasures`, the *function* of a `ListMeasure` is applied in to all measures that defines the list or all those that meet the restriction defined by the attribute *functionRestriction*. For example, the *function = SUM* of a `ListMeasure` can be related to the *restrictionFunction = all*, to indicate that the measure will sum all values of the list members. The *function = COUNT* of another `ListMeasure` related to the attribute *restrictionFunction = greater than 5* indicates that the measure only counts the values of the elements in the list that are greater than 5. The main restriction of a list is that all its elements must be of the same type. For example, we can have a list of external values, a list of time measure values, a list of count measure values, etc. but these types cannot be mixed. The *typeOfList* attribute is used to specify the type of values that make up the list.

Finally, in addition to the restrictions described above for some elements of the metamodel, there are a certain restrictions that must be taken into account when defining the new composite measures. These are detailed below.

- An `ExpandedMeasure` should be linked with at least one measure, but it cannot be linked with another `ExpandedMeasure`. Hierarquical levels can be represented using a combination of `CollapsedMeasures` and `ExpandedMeasures`. An `ExpandedMeasure` may contains as many `CollapsedMeasures` as necessary.
- A `CollapsedMeasure` should have at least one `CompositeConnector`.
- A `TimeConnector` must have a pair of links *(from, to)* that connects it with a `TimeInstantCondition`.
- A `CountConnector` must have a link *when* that connects it with a `TimeInstantCondition`.

- A `StateConditionConnector` must have a link *meets* that connects it with a `ProcessInstanceCondition`.
- A `DataConnector` must have a link *precondition* that connects it with a `DataPropertyCondition`.
- An `AggregatedConnector` must have a link *aggregates* that connects it with a `MeasureConnector`.
- A `ListConnector` must have one of the two links: a *connectsExternalValues* that connects it with an `ExternalValue` provided by users or a *connectsMeasures* that connects it with `MeasureDefinitions`.
- A `ParameterConnector` must *receive* one *ExternalValue*.
- A `CollapsedMeasure` is an instance of an `ExpandedMeasure`, so the number of `CompositeConnectors` in a `CollapsedMeasure` should be the same that the number of measures not-connected in an *ExpandedMeasure*.
- A `ListMeasure` must be connected with at least one base measure or with an `ExternalValue`.
- All elements in a `ListMeasure` must have the same type, either an `ExternalValue` or one type of measure.
- All elements in a `ListConnector` must have the same type, either an `ExternalValue` or one type of measure.

4.5 Abstraction in the Modelling of PPIs using PPINOT Templates

Approaches such as [32, 130] define and model PPIs in a structured way by means of a list of fixed attributes with the aim of reducing ambiguity and the possibility of missing information in PPI definitions. In this section, we extend the PPINOT-Template notation and the linguistic patterns related to them, which were introduced in Section §2.6.1, to incorporate concepts of abstraction described in previous sections. Using this notation we benefit from the advantages provided by PPINOT and its templates: to model PPIs in a fixed form by means of a set of attributes, to maintain traceability with the business process, to make definitions understandable to all stakeholders involved, to maintain the high expressiveness provided by the PPINOT metamodel, and the use of linguistic patterns to provide guidance on the definition of PPIs.

Although in the structure of the PPINOT template for a PPI definition, we can define a PPI using a set of attributes (see Table §2.1), since the metamodel extension is focused on the new measures and the elements related to them (see Figures §4.4 and §4.5), the only attribute that we modify and extend is the *Measure Definition*. To do this, new linguistic patterns are included. Our main interest in this section is twofold: first, to include linguistic patterns to describe `CompositeMeasure` as a double definition that includes `CollapsedMeasure` and `ExpandedMeasure`; and second, to include a linguistic pattern to describe `ListMeasures`. Below we describe the new linguistic patterns and present some examples to clarify how they are used.

4.5.1 Linguistic Patterns for Composite Measures

As we described before, a `CompositeMeasure` can be used as one of two types: `ExpandedMeasure` and `CollapsedMeasure`. Those measures are linked with a set of measures that define them by means of set of `CompositeConnector`. In our proposal of extension of PPINOT templates, we maintain the relationship between those concepts. In this sense, two different linguistic patterns are defined, one for each measure. The concept of `CompositeMeasure` is covered by placeholders used in linguistic patterns. Each placeholder, as in original PPINOT templates, represents literal values (named using lower case) or linguistic patterns (named using upper case first letter). A placeholder may also represent an `ExternalValue`, a `BPElement` or some attributes of measures or `BPElement`.

To illustrate the problem addressed using `CompositeMeasures`, one of the patterns listed in Table §4.1 is used as example. We focused on the "PCT" pattern, which calculates the percentage of a set of values. In this pattern the information can be taken from different sources (data objects, activities, etc.). Specifically, we use the PCT pattern to calculate the percentage using information taken from data objects (hereafter, *Percentage_Data*).

To calculate a PPI and define a measurement using the *Percentage_Data* pattern, it is necessary to know the value of the `DataObject` that provides the information and the state of the `DataObject` to differentiate the information to be taken into account. The total of information gathered will be divided into the total of `DataObjects` registered during the process execution. For example, a PPI whose `MeasureDefinition` indicates how to calculate the number of cancelled orders, identify the total of cancelled orders and then it will be divided by the number of orders registered in a period. Each time a PPI is calculated using a *Percentage_Data* pattern would be necessary to use the following description. Let us assume that PPI is calculated taking information from the `DataObject Order`.

Measure Definition =	the function $\frac{a}{b} * 100$ where a is the sum of the number of *types data object Order* becomes *cancelled*, b is the sum of the number of *times data object Order* becomes *registered*.

In order not to have to define a new measure each time a PPI is defined as a percentage by taking information from a `DataObject`, we propose the use of composite measures. Thus, the measure is defined as an extended measure, which is instantiated by the collapsed measure. This usefulness is more evident in the implementation of PPIs, since using the composite measures the values expected for each measure will already be defined and restricted and the user will have less margin for error. The proposed patterns are defined below.

CompositeMeasure ::= *ExpandedMeasure* | *CollapsedMeasure*

ExpandedMeasure ::=

the [composite] measure <*measure name*> is calculated as {*MeasureForComposite*[($< j_1, ..., j_n >$)[, which can be instantiated as <*description using* $j_1, ..., j_n >$]] [and { <*external value*>}$_i$] }, where $< j_i >$ is a parameter related to *MeasureForComposite*$_i$.

CollapsedMeasure ::=

the [composite] measure <*measure name*> { is calculated using {<*MeasureForComposite type*$_i$ >($< j_1, ..., j_n >$) } $_{i=1,...,n}$
| <*description using* $j_1, ..., j_n >$[, where $< j_i >$ is a parameter.] }

MeasureForComposite ::=

{ *TimeMeasure* | *CountMeasure* | *ConditionMeasure* | *DataMeasure*
| *AggregatedMeasure* | *DerivedMeasure* | *ListMeasure*}

MeasureForComposite type ::=

{ time measure | count measure | condition measure | data measure
| aggregated measure | derived measure | list measure }

In the previous definitions, we consider that an `ExpandedMeasure` can contain one or more `MeasureForComposite` and can also include a set of *ExternalValues*. `Measure-ForComposite` is composed of the linguistic patterns for measures described in Table §2.2, with the difference that in a `CompositeMeasure` they may not be specified because they represent parameters that must be provided when instantiation of the `Collapsed-Measure`. *ListMeasure* will be described in following subsections.

The example about the cancelled orders was defined using a `DerivedMeasure` linguistic pattern. It can also represented using `CompositeMeasure` linguistic patterns. First, the measure should be represented as an *ExpandedMeasure* and then, the same measure can be instantiated using a *CollapsedMeasure* in each process where the PPI has been defined. One way to represent our example using a composite measure is presented below.

Measure Definition (ExpandedMeasure) ::=

the [composite] measure `Percentage_Data` is calculated using as the function $\frac{a}{b} * 100$, where
`a` is the `sum of the` number of times *data object* $<p\_dobject\_name_1>$ becomes $<p\_state_1>$,
`b` is the `sum of the` number of times *data object* $<p\_dobject\_name_2>$ becomes $<p\_state_2>$,
which can also be instantiated as:
"the [composite] measure `Percentage_Data` calculates the percentage of $<p\_dobject\_name_1>$, <p_state$_1$ > from the total of $<p\_dobject\_name_2>$, $<p\_state_2>$".

Note that although the `ExpandedMesure` definition includes the formula for calculating the PPI, there is a set of placeholders that must be maintained, because their actual value will depend on each instantiation using a `CollapsedMeasure`. In the previous definition *p* indicates a parameter of the measure definition, *type, name* and *state* represent attributes of a business process element and *dobject* represents a *DataObject*. For `TimeMeasure` and `CountMeasure`, parameters represent `Events`. This measure can be instantiated as follows:

Measure Definition (CollapsedMeasure) ::=

the measure `Percentage_Data` calculates the percentage of `orders (`*`Order`*`) cancelled (`*`cancelled`*`)` from the total of `orders (`*`Order`*`) registered (`*`registered`*`).`

The same `ExpandedMeasure` `Percentage_Data` can be instantiated more than once. It can be used to define different types of percentages, or even in different business processes representing many situations. *Percentage_Data* can be used to calculate, for example, *orders with errors*, *orders shipped*, or even using a different `DataObject`, e.g. percentage of *invoices paid*. Let us suppose a PPI that needs to calculate the percentage of orders packaged during a period of time. For this PPI its measure definition could be defined using the *Percentage_Data* pattern, and it can be instantiated as follows:

Measure Definition (CollapsedMeasure) ::=

the measure `Percentage_Data` calculates the percentage of `orders (`*`Order`*`) packaged (`*`packaged`*`)` from the total of `orders (`*`Order`*`) registered (`*`registered`*`).`

or, as another example, the percentage of invoices paid, could be defined as follows:

Measure Definition (CollapsedMeasure) ::=

the measure `Percentage_Data` calculates the percentage of `invoices (`*`Invoice`*`) paid (`*`paid`*`)` from the total of `invoices (`*`Invoice`*`) registered (`*`registered`*`).`

4.5.2 Linguistic Patterns for List Measures

A `ListMeasure` is proposed with the aim of being able of managing collection of values by applying a formula over all of them, specifying or not the number of elements involved. In this section, we introduce a linguistic pattern for the `ListMeasure`. Let us suppose a PPI defined to measure the execution time of a set of activities (Activity A, Activity B and Activity C) during a certain period of time, and that those activities are not necessarily sequentials. This PPI could be applied in many business processes, but in each of them, a different number of activities may be involved.

Using a `ListMeasure` a funtion is applied over a set of fixed or variable measures or external values, such as numbers, strings, etc. In a general way, a `ListMeasure` can be defined as a `DerivedMeasure`, explicitly indicating the type and number of values required to calculate it as is shown below.

ListMeasure ::=

> the [list] measure <*measure name*> applies a function list { *sum* | *maximum* | *minimum* | *average* | ... } over $< x_1, ..., x_n >$, where
> $< x_i >$ is {TimeMeasure | CountMeasure | ConditionMeasure | DataMeasure | <*external value*>} }

The example about measuring the execution time of a set of activities can be defined as follow:

ListMeasure ::=

> the list measure `Total_execution_time` applies a function *sum* over `a,b,c`, where
> *a* is the *duration between the time instant when* `Activity A becomes active` *and when* `Activity A becomes finished`,
> *b* is the *duration between the time instant when* `Activity B becomes active` *and when* `Activity B becomes finished`,
> *c* is the *duration between the time instant when* `Activity C becomes active` *and when* `Activity C becomes finished`.

However, with the aim of facilitating reuse of a `ListMeasure`, another definition is proposed based on the concept of `CompositeMeasure`. The first definition describes the content of the list and the function applied over all its elements, leaving open the possibility to relate as many measures to a list as necessary; while the other definition instantiates the list, specifying the business process elements or external values to be used in the calculation of the measure for a particular case. The first definition is called `GeneralListMeasure`, as in the case of `CompositeMeasure` definition, parameter of measures are used to facilitate reusability of list definitions, as is shown below.

GeneralListMeasure ::=

[the [list] measure <*measure name*> applies] a function list { *sum* | *maximum* | *minimum* | *average* | ... } over {
[<n>] <*MeasureForList type*>[s], defined as $\{< MeasureForList_i >\}_{i=1,...,n}$
| [<n>] <*external_value_type*>[s], defined as $\{< external\ value_i >\}_{i=1,...,n}$ }

MeasureForList ::=

{ *TimeMeasure* [over ({
$<BP\ element\ type_a>$, $<BP\ element\ name_a>$, $<BP\ element\ state_a>$,
$<BP\ element\ type_b>$, $<BP\ element\ name_b>$, $<BP\ element\ state_b>$
| $<BP\ event\ name_a>$, $<BP\ event\ name_b>$ })]
| *CountMeasure* [over ({
<*BP element type*>, <*BP element name*>, <*BP element state*>
| <*BP event name*> })]
| *ConditionMeasure* [over ({
<*BP element type*>, <*BP element name*>, <*BP element state*>
| [<*data object state*>] <*data object name*> <*condition on data object property*> })]
| *DataMeasure* [over (<*data object property name*>, <*data object name*>)] }

MeasureForList type ::=

{ time measure | count measure | state condition measure | data measure }

For our particular example presented at the beginning of the section, leaving open the possibility to use this measure difition in different business processes. `GeneralListMeasure` can be used in the following way.

GeneralListMeasure ::=

the list measure `Total_execution_time` applies a function list *sum* over <*n*> `time measures`, defined as
$\{\text{TimeMeasure}_i(<p\_type_{ia}>, <p\_name_{ia}>, <p\_state_{ia}>, <p\_type_{ib}>, <p\_name_{ib}>, <p\_state_{ib}>)\}_{i=1,...,n}$

The same measure can be defined using a specific number of measures, as follows:

GeneralListMeasure ::=

the list measure `Total_execution_time` applies a function list *sum* over 3 `time measures`, defined as
$\text{TimeMeasure}_1(<p\_type_{1a}>, <p\_name_{1a}>, <p\_state_{1a}>, <p\_type_{1b}>, <p\_name_{1b}>, <p\_state_{1b}>)$,
$\text{TimeMeasure}_2(<p\_type_{2a}>, <p\_name_{2a}>, <p\_state_{2a}>, <p\_type_{2b}>, <p\_name_{2b}>, <p\_state_{2b}>)$,
$\text{TimeMeasure}_3(<p\_type_{3a}>, <p\_name_{3a}>, <p\_state_{3a}>, <p\_type_{3b}>, <p\_name_{3b}>, <p\_state_{3b}>)$,

In the first example, `GeneralListMeasure` definition can be used with 1 to n measures in a list; but in the second example, it is possible to reuse the measure if thee time values are involved. Both definitions can be used to represent our example related to execution time of activities A, B and C. The instantiation of a `GeneralListMeasure` is made by means of a `ListMeasure` definition.

Measure Definition (ListMeasure) ::=

> the [list] measure <*measure name*> is calculated over <*MeasureForList type*[s]> with {(< $j_1,...,j_n$ >) [, where < j_i > is parameter of <MeasureForList$_i$ >]}$_{i=1,...,n}$

This linguistic pattern can be used in following way.

ListMeasure ::=

> the list measure `Total_execution_time` is calculated over `time measures` with *(Activity,A,active,Activity,A,finished)*, *(Activity,B,active,Activity,B,finished)*, *(Activity,C,active,Activity,C,finished)*,

Let us suppose another scenario where the same PPI is calculated using a different set of activities. In this case we also have 3 time values for the list, and we calculate time values using groups of activities, where some of those activities are sequential. For example, we have the execution time of *Activities A, B* and *C*, and they are executed as a sequence into the business process. Then we have the execution time of *Activity E*; and finally, the execution time of another sequence formed by *Activities R* and *S*. Any of `GeneralListMeasure` allows us to define the measure in the following way.

ListMeasure ::=

> the list measure `Total_execution_time` is calculated over `time measures` with *(Activity,A,active,Activity,C,finished)*, *(Activity,E,active,Activity,E,finished)*, *(Activity,R,active,Activity,S,finished)*,

If the calculation of a `ListMeasure` value needs to consider more or less than three time values, the measure needs to be defined using the first `GeneralListMeasure` or we need to modify the second general definition and specify more or less elements in the list.

4.6 Abstraction in the Modelling of PPIs using Visual PPINOT

The representation of concepts by means of graphic models often makes those models easier to understand. However, according to [110], humans have difficulty understanding all elements that are part of a diagram composed of many elements, especially when users/readers are not directly associated with the context depicted.

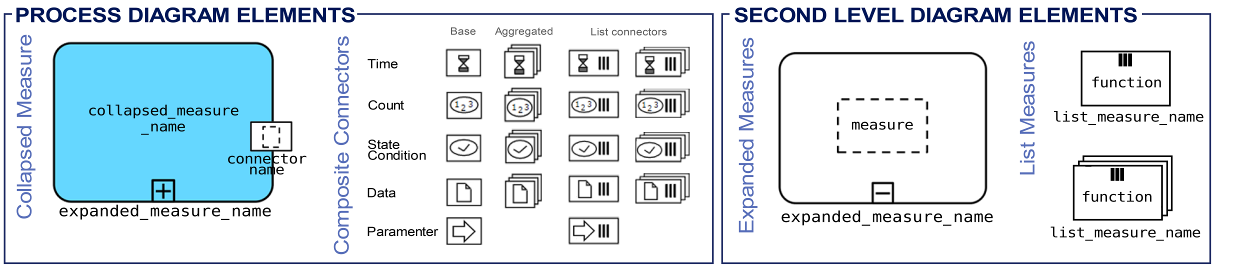


Figure 4.6: Extension of Visual PPINOT including Abstraction concepts

The principle of *Complexity Management* described in [110] refers to the ability of a visual notation to represent information without overloading the human mind. In this context, the author describes complexity as a diagrammatic complexity which is measured by the number of elements included in a diagram. Modularisation and hierarchical structures (levels of abstraction) are presented as mechanisms to manage complexity.

In order to illustrate the advantages of reusing PPIs using abstraction concepts introduced in previous sections, in this section we extend the Visual PPINOT notation, the graphical notation of PPINOT. This extension is based on restrictions included in the PPINOT metamodel. The new measuring elements are shown in Figure §4.6. Those elements enable the implementation of hierarchical levels, focused on the measure definition.

At first glance we can identify two levels. In the first one, several measures are hidden using a `CollapsedMeasure`. This measure can interact with original Visual PPINOT elements (see Section §2.6.2). The second level is conformed by the explicit representation of all PPINOT measures required to define a composite measure and that were hidden in the first level. These measure definitions are contained in an `ExpandedMeasure`. An `ExpandedMeasure` can also contain other `CollapsedMeasures`. In Figure §4.6, the new elements of Visual PPINOT are classified according those two levels and are described below.

- **Process Diagram Elements**: These new elements can be used in the same diagram that original Visual PPINOT elements.
  - `CollapsedMeasure`: Represents an abstraction of a set of measures that are not explicitly incorporated in diagram level of PPI definitions. This measure indicates the use of pattern that is being reuse. A `CollapsedMeasure` uses two names, one to identify it in a unique way, and the other one to indicate the pattern (definition made in a expanded measure) that is being used. This measure is connected with other elements of the diagram (business process elements or measures) by means of `CompositeConnector`. A `CollapsedMeasure` may contain as many connectors of any type as necessary.
  - `CompositeConnector`: This element is used to indicate from which business process elements or measures a `CompositeMeasure` obtains information. In

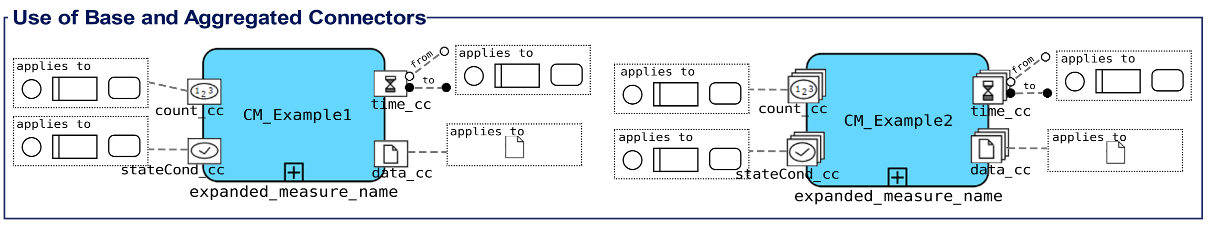


Figure 4.7: Examples of how to use `BaseConnector` and `AggregatedConnector`

the graphical notation we classified `CompositeConnector` depending on the elements that it connects. In accordance with the metamodel presented in Figures §4.4 and §4.5, four groups of connectors are identified, three related to measures (`BaseConnector`, `AggregatedConnector` and `ListConnector`) and one related to external values provided by users (`ParameterConnector`). All those types of connectors have an associated *name* to distinguish them from others of the same type.

* `BaseConnector` is used to connect a `CompositeMeasure` with business process elements. There are four types of `BaseConnector`, one for each `BaseMeasure`: `TimeConnector`, `CountConnector`, `StateConditionConnector` and `DataConnector`. The first three can be connected to business process activities, events or pools of a business process model. The `DataConnector` can only be used to measure a `DataObject`. Each type of connector represents the measure that is going to receive the information taken. For example a `TimeConnector` connected with an activity indicates that the pattern represented by the `CompositeMeasure` that contains the connector has a time measure in the `ExtendedMeasure` related to it. Figure §4.7 shows business process elements to which `BaseConnector` can be connected.
* `AggregatedConnector`. Similar to the previous type of connector, it connects a `CompositeMeasure` with business process elements and can be instanciated by one of the four types, one for each base measure. The difference between a `BaseConnector` and an `AggregatedConnector` is that the last one represents and `AggregatedMeasure` of the type represented by the connector instead of a `BaseMeasure`. Figure §4.7 shows business process elements to which `AggregatedConnector` can be connected.
* `ListConnector`. This type of connector can be instantiated in several ways. A `ListConnector` indicates that a set of elements are required. All elements must be of the same type and each one represent a type of `BaseMeasure`. We can also indicates if an aggregated function is going to be applied over the set of values related to the list. In this case, we also have four types, but the icon depicts three rectangles. Figure §4.8 shows how a `ListConnector` should be use with a specific type of measures, depending on each type of connector and that all measures of a

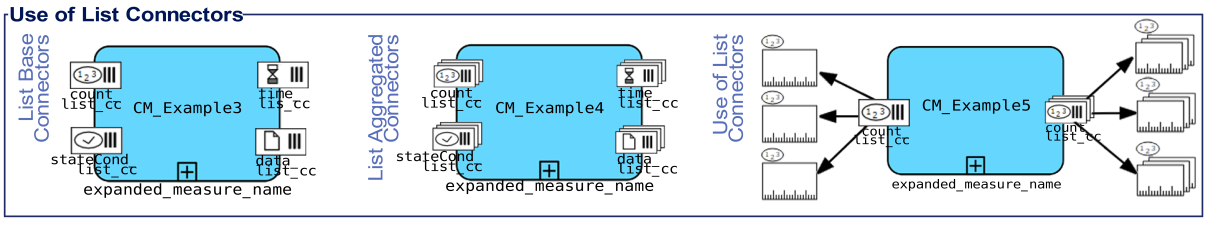


Figure 4.8: How to use a `ListConnector`. The use of `BaseConnector` is shown to the left of the figure. The use of `AggregatedConnector` is shown in the middle of the figure. All elements in a list must be of the same type as shown on the right of the figure.

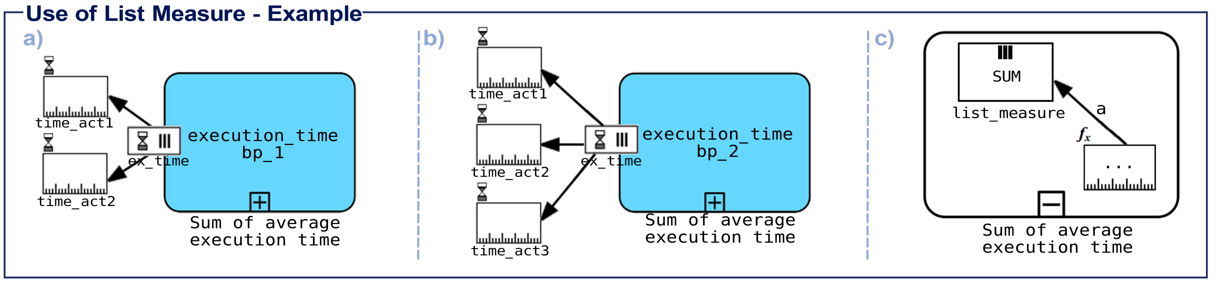


Figure 4.9: Example of reuse using `ListConnector`. *a)* A `ListConnector` connected with two measures; *b)* A `ListConnector` connected with three measures; *c)* Excerpt of a PPI definition using a `ListMeasure`.

list should be of the same type. `ListConnector` aggregated or not can be used and combined in the same `CompositeMeasure` as better suits and and including any number of connectors of each type. Regardless of the type of list connector used, its importance lies in the ease of reuse of the *CompositeMeasure* definition. For instance, the SCOR measure *Sum of the average execution time* of several activities of a business process can be calculated with 2, 3 or more time of tasks, depending on how many tasks the business process has (Figure §4.9, sections *a* and *b*). A `Collapsed-Measure` that models this SCOR measure should contain a `ListConnec-tor`. Depending on the business process where the `CollapsedMeasure` is used, two, three or more measures can be connected to measure, but the original definition is the same for both of them (See Figure §4.9, section *c*). This functionality lets define the PPI or pattern only once for all business processes that require a similar measure functionality.

* `ParameterConnector` is used to indicated that an `ExternalValue` is required by the measure. A `ParameterBaseConnector` indicates a single external value is required, while a `ParameterListConnector` indicates a set of external values are required.

- **Second Level Diagram Elements**: This level is represented by means of `Expand-edMasure`.

  – `ExpandedMeasure`. It is used to depict in an explicit way, all PPINOT mea-

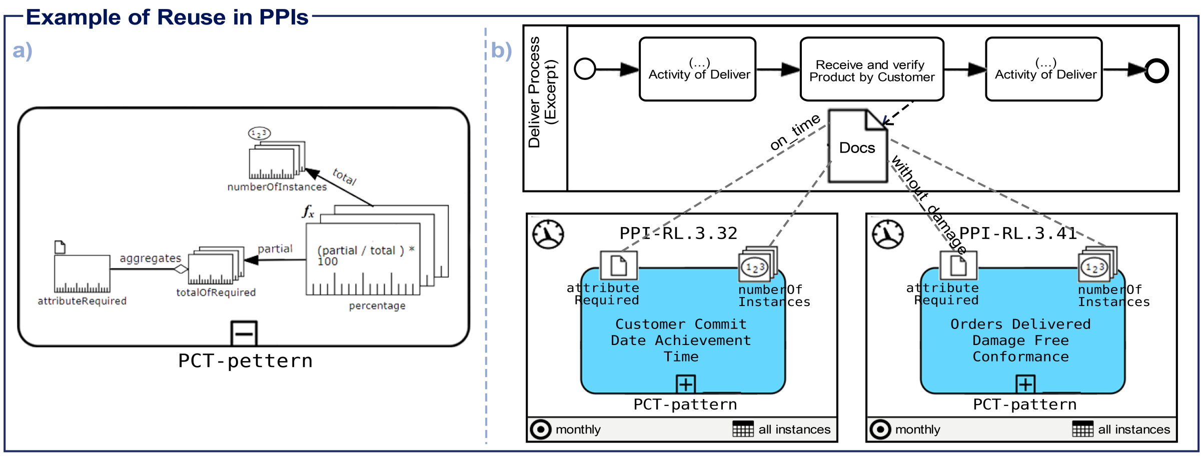


Figure 4.10: Example of a pattern defined in an ExtendedMeasure and its reuse in two PPIs.

sures required to calculate a PPI and that were hidden in a CollapsedMeasure in the upper level diagram. The ExpandedMeasure acts as a container of a set measures. The *name* attribute uniquely identifies the pattern to be reused that is described in it. To be used and instantiated, at least one CollapsedMeasure should have the same name of an ExtendedMeasure. Each CompositeConnector in a CompositeMeasure should be represented in this detailed diagram inside an ExpandedMeasure by means of a BaseMeasure or an AggregatedMeasure depending on the type of connector. If a ListConnector was used, a ListMesure should be contained into a ExpandedMeasure.

– ListMeasure. It is included among the second level elements because it can only be used inside a ExpandedMeasure. Measures that make up a ListMeasure must all be of the same type. A ListMeasure has two attributes: *name* and *function*. The attribute *name* helps us to uniquely identify each measure, while *function* represent the function to be applied to all elements of the list.

As an example of benefits that can be obtained from the reuse of *CompositeMeasures* and *CompositeConnectors*, and the use of the extension of Visual PPINOT, we present an example based on one of the patterns identified in table §4.1 identified in the SCOR model. The *PCT* pattern was identified 76 times in the SCOR measures. If we want to model all SCOR measures without using the Visual PPINOT extension, we should model each one of them one by one. This would include a large number of redundant models. If we use the proposed Visual PPINOT extension, we only need to model the detail of the measure once (within a second level diagram that includes the ExpandedMeasures) and instantiate this measure as many times as we need it, to connect it with the business process elements that interest us. This scenario is shown in Figure §4.10. In section *a* of that figure, the ExpandedMeasure is created to model the content of the patter *PCT-Percentage calculated over a set of values*. Sections *b* of the same figure shows two different instantiations of the same ExpandedMeasure using a business process model. Each instantiation of the pattern is model as a measure definition of a PPI. The first PPI *RL.3.32 - Customer Commit Date Achievement Time* calculates the Percentage of orders which is delivered without damage and it takes information from the activity *Receive and verify Product by Customer* from the three variants of the Deliver process. The second PPI, *RL.3.41 - Orders Delivered Damage Free Conformance* represents the percentage of orders which is received on time as defined by the customer. This PPI is applied to the same activities of the same Deliver process.

4.7 Discussion

In the problem illustration (Section §4.2), reuse in PPI definition was proposed and justified in the context of the SCOR model. Table §4.1 shows a first classification of SCOR measures according to the purpose and expected value of each measure, obtaining six categories, of which five are possible patterns of SCOR measures. After the modelling of several SCOR measures using the two PPINOT notations extended considering abstraction concepts, the first classification proposal has been modified and extended. Table §4.2 shows a new classification of SCOR measures composed of nine categories, of which eight of them are patterns and the other one (*General_Function*) groups together all those measures that cannot be considered patterns because of their variable characteristics and lack of repetition. Each pattern represents a set of SCOR measures that have a very similar functionality and/or structure to define how they must be calculated and whose inputs varies from one definition to another.

The original category *SUM*, included in Table §4.1, is divided in two, because to measure time (*Sum_Time*), a *TimeMeasure* must be use; and to calculate costs (*Sum_Cost*) a *DataMeasure* should be used to acquire the information from a data object of the process. *Sum_Cost* and *Sum_Cost_List* were divided because, for the first case the number of cost values is fixed, whilst for the second case the number of cost values differs depending on the business process where the PPI is defined. The same applies to *Average_Time* and *Average_Time_List*, were the first has a fixed number of time values and for the second the number of time values varies depending on the business process where the PPI is defined.

Relating these categories to the abstraction concepts presented in previous sections we can say that each category can be represented using a `CompositeMeasure`. Specifically, each category (pattern) can be modelled using an `ExpandedMeasure` and it can be instantiated using a `CollapsedMeasure` for each definition required in a particular business process model. According to this analysis, 253 measures may be modelled using composite measures, which represent the 89.7% of SCOR measures. In addition, 65 measures may be defined using a *ListMeasure*, representing the 23% of SCOR measures.

In Appendix §B, we have included several examples of reuse PPI definitions using extensions of the PPINOT Templates and linguistic patterns and the PPINOT tool. In Appendix Section §B.1, all patterns presented in Table §4.2 have been modelled us-

Table 4.2: New general patterns identified in SCOR

Pattern	**Description**	*N*° **Measure**
Sum_Time	Sum of a set of time values.	8
Sum_Cost	Sum of instance values expressed as money.	10
Sum_Cost_List	Sum of a set of values expressed as money.	30
Average_Time	Average calculated with instances of time values.	89
Average_Time_List	Average of a set of time values.	25
Percentage_Data	Percentage calculated with data values (one condition).	64
Complex_Percentage	More than one condition must be met to define the instances that make up the numerator of the percentage formula.	10
Number	Counting of conditions	9
Function_5	A predefined SCOR function is applied. 5 point rolling average is calculated with four previous quarters values and one projection.	8
General_Function*	A particular formula is calculated over a set of values. (No pattern)	29
Total of measures		***282***

* This is not a pattern. This is a set of measures that does not correspond to another category.

ing linguistic patterns of PPINOT. When modelling those patterns we can identify that there are some of them that are very similar to each other, such as patterns *Average_Time* and *Sum_Time*. As explained in the appendix, these two patterns could be unified and defined as a single pattern, as long as the *function* required to define the `DerivedMeasure` is expressed as a parameter of the measure. From this example we can deduce that the way of modelling and defining patterns is not unique and that the characteristics and inputs of each pattern must be defined to facilitate the modelling of measures and PPIs in the specific scenarios where they will be used.

In Appendix Section §B.2, we explain how the PPINOT core tool has been extended as a proof of concept to analyse PPINOT models taking into account abstraction concepts in their definitions. In this sense, the Appendix Section §B.3 shows an example of modelling composite measures using the extended core of PPINOT. From this example we can conclude that the use of patterns modelled using *composite measures* greatly reduces the source code needed to model them, compared to the modelling of measures one by one without the use of patterns.

With regard to the research question of reuse (R-RQ) presented at the beginning of this article, we can say first, that abstraction is a good alternative to implement reuse in PPI definitions because help us to reduce details that can confuse users, especially to non-experts in the field, due to the large amount of information provided. In this sense, we propose a set of *composite measures* and connectors to provide at least two hierarchical levels in PPI definition.

Although in the examples given throughout this chapter we have only considered two levels of modelling in PPI definition (one at the level of the business process elements, where the `CollapsedMeasures` are included and the other one where the pattern is detailed by means of an `ExpandedMeasure`), the elements and constraints of the metamodel allow us to relate composite measures to generate more levels. For example, using two composite measures in the definition of a PPI does not create a new level, but if the `ExpandedMeasure` contains an `CollapsedMeasure` in its definition, a new level of abstraction would be added.

It is important that, like the sub-processes in business process models, it is necessary to maintain a balance between the levels of abstraction and the complexity of the model, since if too many levels of depth are defined in the definition of a PPI, traceability may be more difficult to identify or the user may identify the model as too abstract because of the lack of too much information, generating the opposite effect of what is sought with abstraction.

In addition to extension of the metamodel to indicate new components and constraints related to the integration of abstraction concepts, we have used PPINOT notations to propose different modelling alternatives of reuse in PPI definitions. First, templates and linguistic patterns were extended to allow us to define PPIs in a structured way, but with flexibility to use information that is not taken from the process, but from `ExternalValues`. And second, the extension of the graphical notation can be useful for non-technical users to be able to transmit their ideas in accordance with certain restrictions, but without the need to delve into technical details of modelling or definition. Finally, although the PPINOT core extension is still a very preliminary version of the tool extension, we consider it is important because it is the first step to extend the automatic computation and analysis of features provided by PPINOT, allowing us to be able to verify whether the models provided are correct.

4.8 Summary

In this chapter, reuse is applied to the definition of PPIs by means of abstraction concepts. Based on the SCOR model, we identified patterns of measures that can be applied to facilitate the definition of PPIs. The PPINOT metamodel and its notations have been extended to illustrate how those concepts can be used in the modelling of PPIs. Although several examples of modelling SCOR patterns using PPINOT extensions have been included, Appendix §B contains more examples using PPINOT template extension and PPINOT core tool extension.

Variability in PPI Definitions

"Never think that lack of variability is stability."

Nassim Nicholas Taleb (1960),
Lebanese-American essayist, scholar, statistician, former trader an risk analyst

Changes and variations between business processes can be reflected in different process perspectives. Although there are proposals focused on managing variability in perspectives of business processes such as the control-flow, data or resources, unfortunately, as far as we know, there are not proposals to deal with the management of variability in the performance perspective of business processes. In this chapter we address this topic. Section §5.1 introduces the chapter. Section §5.2 describes an illustration scenario where variability is reflected. In Section §5.3, we identify dimensions of changes related to variability in business processes and in PPIs themselves. Section §5.4 indicates how those dimensions of changes can be represented with PPINOT. In Section §5.5, we show how the extension of PPINOT can be used in conjunction with two Variability Business Process Modelling Languages. Section §5.6 describes how variability is reflected in two real scenarios. A discussion about this chapter is presented in Section §5.7. Finally, the chapter is summarised in Section §5.8.

5.1 Introduction

A business process can be applied in different scenarios, but in many cases it needs to be adapted to comply with different requirements such as regulations found in different countries and regions [63], due to changes in original process requirements [61], to reflect new allocation of responsibilities, new strategic and business goals or by changes in general inputs of the business process [107]. In many cases, the original business process and all adaptations generated from it, need to be managed as a collection [106]. This collection, in which each alternative meets a common goal but is implemented in a different way, is called a business process family, hereinafter *process family* (PF) for short [61, 142] and each member of the PF is called *process variant* (PV).

There are several criteria for defining when a business process is considered a process variant of another process. Certain authors consider a process variant to be those processes that share business rules and/or objectives [168], those that have variations in inputs and business goals [107] or those that have similar inputs and outcomes [106]. In [168], variability is defined as "a technique for designing business process(es), in which business rules and/or objectives are similar to one another in some ways but different in others, by factorising the similarity(ies) and grouping the difference(s) into a business process of reference". In this chapter, we consider all those ideas to identify the variants of a process if each possible variant meets at least one of the characteristics previously described.

Changes and variations between business processes can be reflected in different process perspectives, not only in control-flow, but also in data or resources [88]. One of them is the performance perspective of business processes which is concerned with the definition of performance requirements, addressing different performance dimensions, such as time, cost and quality [95], and it is usually expressed as a set of Process Performance Indicators (PPIs).

Although there are many proposals focused on managing the variability in different perspectives of business processes (see Section §3.2.2) and others focused on measuring the performance of business processes (see Section §2.5), unfortunately, as far as we know, there are not proposals to deal with the management of variability in the performance perspective of business processes. Up to know, PPIs defined over processes are managed as independent definitions, even if PPIs are defined for several process variants, making the management of variability a repetitive, laborious and error-prone task. In contrast, having an explicit proposal to manage the variability of PPIs together with other perspectives of business processes helps to guarantee consistency and correctness across PPI variants and can reduce maintenance efforts and cost of changes.

In this chapter, we analyse how variability in the business process affects the performance perspective from the definition of PPIs. The extension of the PPINOT metamodel is the basis for defining a proposal that allow the definition and modelling of PPIs considering variability of business process models and PPIs themselves. This proposal is integrated with two traditional approaches for managing variability in business processes (C-iEPC[88] and PROVOP[63]). Data presented in this chapter was taken from the incident management process of the Andalusian Health Service and from the processes and metrics proposed in the SCOR model (See Appendix §A).

It should be noted that this chapter is an extension of conference paper [45] where the first ideas related to variability and performance measurement were presented. However, while this chapter includes the original ideas of that contribution, significant changes have also been made to that contribution.

5.2 Problem Illustration

Here we describe an illustration scenario that is based on processes and metrics defined for a process reference model for the supply chain management. The *Supply Chain Operation Reference* model (SCOR) [6] enables users to address, improve, and communicate supply chain management practices within and between all interested parties in the enterprise. We focus on two elements of its structure: *processes* and *measure definitions* (called *metrics* in *SCOR*).

SCOR processes identify a set of unique activities within a supply chain. These activities are described at a high level of abstraction because implementation of processes requires internal and specific definitions of activities of each organisation, which are out of the scope of SCOR. *SCOR measure definitions* are defined as a standard for measuring the process performance.

Due to its structure and the definition of its components, *SCOR processes* have variability. *Deliver process (D)*, for instance, is defined as the processes associated with performing customer-facing order management and order fulfillment activities. It can be implemented in four different ways depending on the selected strategy: *D1-Stocked Product (PV-1), D2-Make to Order Product (PV-2), D3-Engineering to Order Product (PV-3)* and *D4-Retail Product (PV-4)*. Each of them is a process variant of the Deliver process, which are shown in Figure §5.1. Those process variants have a set of common tasks among them, but also have differences depending on the strategy selected. PV-2 varies in 13% with regard to activities defined for PV-1, PV-3 and PV-4 differ in 33% and 100% respectively. For simplicity, we only focus on the three first process variants, because *D-4* is totally different from the other ones.

Variability is also reflected in SCOR through its *measure definitions*, (i) due to their dependence on the business process flow in which they are defined or (ii) by specific requirements of the measures defined for each variant. Measures like *RS.3.120 Schedule Installation Cycle Time* reflect the first case. The measure is defined only in one process variant, because it is connected to the task *D3.4 Schedule Installation* that only appears in PV-3. The second case is manifested in measures that vary regarding the required components to calculate its value. For example, in PV-1 and PV-2 the *RS.2.1 Source*

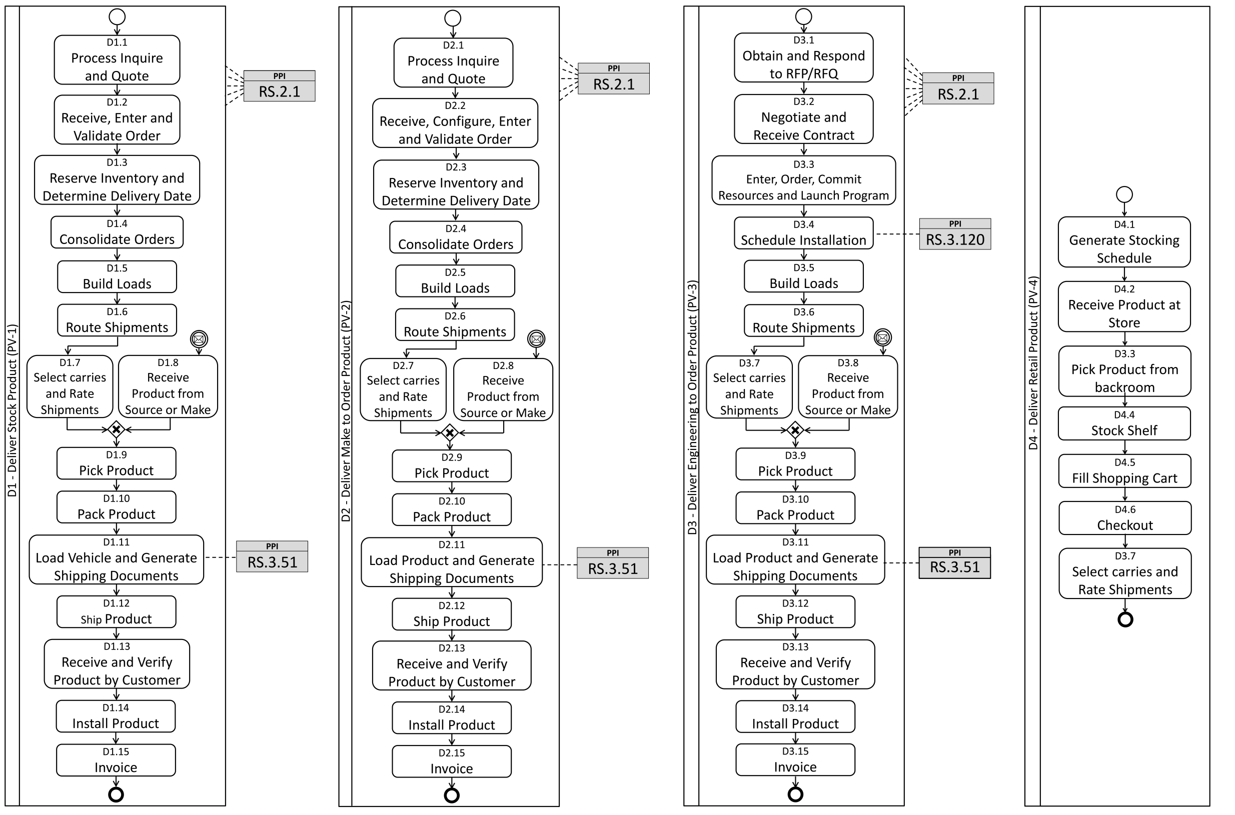

D1 - Deliver Stock Product (PV-1)
D1.1 Process Inquire and Quote
D1.2 Receive, Enter and Validate Order
D1.3 Reserve Inventory and Determine Delivery Date
D1.4 Consolidate Orders
D1.5 Build Loads
D1.6 Route Shipments
D1.7 Select carries and Rate Shipments
D1.8 Receive Product from Source or Make
D1.9 Pick Product
D1.10 Pack Product
D1.11 Load Vehicle and Generate Shipping Documents
D1.12 Ship Product
D1.13 Receive and Verify Product by Customer
D1.14 Install Product
D1.15 Invoice
PPI RS.2.1
PPI RS.3.51
D2 - Deliver Make to Order Product (PV-2)
D2.1 Process Inquire and Quote
D2.2 Receive, Configure, Enter and Validate Order
D2.3 Reserve Inventory and Determine Delivery Date
D2.4 Consolidate Orders
D2.5 Build Loads
D2.6 Route Shipments
D2.7 Select carries and Rate Shipments
D2.8 Receive Product from Source or Make
D2.9 Pick Product
D2.10 Pack Product
D2.11 Load Product and Generate Shipping Documents
D2.12 Ship Product
D2.13 Receive and Verify Product by Customer
D2.14 Install Product
D2.15 Invoice
PPI RS.2.1
PPI RS.3.51
D3 - Deliver Engineering to Order Product (PV-3)
D3.1 Obtain and Respond to RFP/RFQ
D3.2 Negotiate and Receive Contract
D3.3 Enter, Order, Commit Resources and Launch Program
D3.4 Schedule Installation
D3.5 Build Loads
D3.6 Route Shipments
D3.7 Select carries and Rate Shipments
D3.8 Receive Product from Source or Make
D3.9 Pick Product
D3.10 Pack Product
D3.11 Load Product and Generate Shipping Documents
D3.12 Ship Product
D3.13 Receive and Verify Product by Customer
D3.14 Install Product
D3.15 Invoice
PPI RS.2.1
PPI RS.3.120
PPI RS.3.51
D4 - Deliver Retail Product (PV-4)
D4.1 Generate Stocking Schedule
D4.2 Receive Product at Store
D3.3 Pick Product from backroom
D4.4 Stock Shelf
D4.5 Fill Shopping Cart
D4.6 Checkout
D3.7 Select carries and Rate Shipments

Figure 5.1: Example of process variants - Four *Deliver* SCOR processes.

Cycle Time measure requires 5 different time values from 5 tasks of the business process, while in PV-3 this measure requires 7 different time values.

Currently, although there are business process modelling languages (BPMLs) that allow the modelling of variability of business processes, to the best of our knowledge, there are not tools and techniques to model variability in PPIs. In SCOR, for example, Deliver process defines 100, 96 and 96 measures for PV-1, PV-2 and PV-3 respectively, and almost half of them are repeated for all or several process variants. If we want to model them, it would be necessary to model independently the PPIs of each variant, making it a laborious and time-consuming task. Furthermore, if in the future, a PPI changes, we must modify one by one each variant involved, which does not ensure the PPI integrity through all variants, because we could forget to make some changes. If these errors are not detected, they may be carried out throughout the whole lifecycle process leading to new problems like monitoring poorly defined PPIs and collecting inaccurate information that will be used in decision-making, to name a few.

In summary, modelling the variability in PPIs brings similar advantages than modelling the variability in the other perspectives of the business process. Consequently, PPIs should be defined by means of tools and techniques that allow us to represent variability aspects in the business process performance perspective, taking into consideration all the dimensions that affect their variability.

Considering the scenario described above and the possible drawbacks identified, we have formulated a set of questions to help guide research on the relationship between the variability of the business process and the measurement of its performance.

V-RQ1 *What types of variability should be taken into account in the performance measurement of business processes?*

V-RQ2 *How variability can be reflected and modelled from the performance perspective of business processes by means of PPI definitions?*

V-RQ3 *How can the alternatives for modelling PPIs be integrated with current alternatives for the management of variability?*

5.3 Variability in PPI Definitions

As with business process variability, there is no a single criterion for deciding when one PPI should be considered as one variant of another and when it should be considered as a fully independent PPI of another. In order to identify variability in PPIs, we studied several cases of variability in business processes and analysed different models to represent PPIs. First, we modelled the SCOR processes with all their process variants. Then, we selected those processes with more similarities between activities in the control flow of their process variants: *Deliver* and *Make*. After that, we modelled, compared and classified measures defined for those process variants in the SCOR model. Finally, we compared all PPI attributes among process variants, to identify cases of variation on PPIs. A similar study was made for PPIs of IT management processes of the *Andalusian Health Service*.

In this proposal we consider a PPI is a variant of another PPI when both have the same business goals and when (i) the PPI is defined in the same way for some of the process variants of a process family, that is, the values of their attributes are the same for all variants to which it applies; or (ii) when the PPI is defined for all or several process variants, but at least one of the values of its attributes changes from one process variant to another. We called them, *dimensions of change* in PPI definition. The structure of dimensions of change for the PPI variability is shown in Figure §5.2. In it, we can see that Dim-2, related to changes in PPI attributes has four sub dimensions, one for each PPI attribute that can vary in a PPI definition; and from one of them, the measure definition, two other sub dimensions are derived.

Suppose a business process family of more than one process variant. If a PPI is defined for all those process variants and all its attributes are set with the same values for all variants, there is no variability. Instead, if a PPI is defined in some of its process variants but not for all of them, we are representing variability expressed by **Dim-1**, regardless of whether their attributes change or not. In our example in Figure §5.1, if we assume a process family formed by PV-1, PV-2 and PV-3, the PPI *RS.3.51* has not

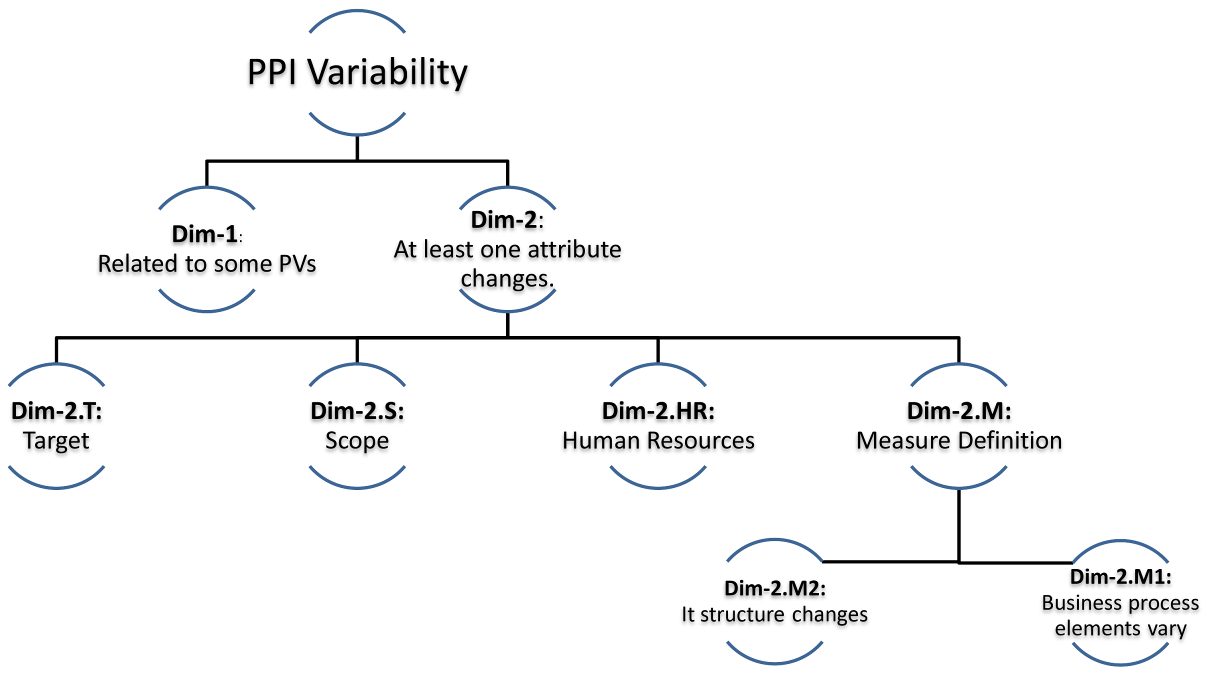

PPI Variability
Dim-1: Related to some PVs
Dim-2: At least one attribute changes.
Dim-2.T: Target
Dim-2.S: Scope
Dim-2.HR: Human Resources
Dim-2.M: Measure Definition
Dim-2.M2: It structure changes
Dim-2.M1: Business process elements vary

Figure 5.2: Dimensions of change related to PPI definitions

variability according to **Dim-1** because it appears for all PV on the PF. The PPI *RS.3.120*, however, has variability according to **Dim-1**.

In addition, a PPI, regardless of the behaviour derived from **Dim-1**, may vary depending on the changes applied over the value of one or more of its attributes, which is related to the definition of **Dim-2**. In Section §2.5 we mentioned attributes that conform a PPI, and here we list some cases where the PPI variability is reflected, considering that a PPI varies if at least one of the following attributes changes:

Target *(T)* changes when the target value to be reached changes. For example, the *Andalusian Health Service* defines a PPI for measuring the percentage of resolved incidents in a period of time and in which its target values depend on the priority established for the measured service. If priority is very high, the target value is very high (resolved incidents $>=$ 95%); if priority is high, the target value changes (resolved incidents $>=$ 90%) and if priority is normal, the target value also changes (resolved incidents $>=$ 82,5%).

Scope *(S)* changes when the set of instances to be evaluated changes. For example, if we have one PV that applies during weekdays and another one that applies in weekends (e.g., due to limited availability of resources available on weekends), we might define two variants of the same PPI, one that evaluates instances that take place on weekdays, and another one that evaluates those that take place on weekends.

Human resources *(HR)* may change by two attributes: **responsible** and **informed**.

For example, taking up the previous example, depending on the priority of an incident, the person responsible for the PPI or the person informed about its value might change, e.g., because high priority incidents are resolved by a different team.

Measure definition ***(M)*** is a complex attribute through which a PPI is calculated. In this case, there are two dimensions of change, one related to the measure definition itself and another one related to the relationship with the business process:

Dim-2.M1: *A measure definition maintains its structure, but may vary depending only on the business process elements to which it is connected.*

Dim-2.M2: *A measure definition changes its structure and may vary depending on the requirements of the process variant*.

When we talk about the structure of a measure definition, we are talking about the set of intermediate logical-mathematical operations required to define a complete measure. For example, a measure definition can be specified as the sum of three values, *a*, *b* and *c*; however, each of those values can be single values taken from the process such as data of a data object, or they can be more complex, such as the sum of several values, the maximum or minimum value of a data set or even the result of a complex function defined according to organisation requirements.

Dim-2.M1 might occur when a PPI is connected to a business process element, such as a task, that is not available for all PVs where the PPI is defined, or because for certain PVs requirements of the PPI definition changes and the PPI is assigned to a different task depending on the PV where the PPI is defined. An application example of **Dim-2.M1** is the PPI defined over the SCOR measure *RS.3.51 - Load Product & Generate Shipping Documentation Cycle Time*, which is defined in the Deliver process over task 11. In *PV-1* this PPI is computed over task *D1.11 Load Vehicle & Generate Shipping Documents*, but in *PV-2* and *PV-3* this task is not available (See Figure §5.1). For this reason the same PPI is defined over an equivalent task, *(D2.11, D3.11) Load Product & Generate Shipping Docs*.

Dim-2.M2 might occur when a PPI is defined in two PVs (or more) as the sum of some measures, but in PV-1 needs to explicitly use a set of measures that differs from the set of measures defined for PV-2. An example is the *RS.2.1 Source Cylce Time* measure definition (see Figure §5.1) that is defined in PV-1, PV-2 and PV-3, but in PV-1 and PV-2 the measure requires information from 5 activities, while in PV-3 it requires information from 7 activities.

5.4 Variability in PPINOT

In Section §2.6, we described PPINOT as a metamodel that allows the definition and modelling of a *performance model* composed of a set of *PPIs*. In the original version of PPINOT [31] a PPI is related to a single business process. A PPI definition is made by means of a set of attributes. In this section, we describe how PPINOT has been modified to include the dimensions of change previously presented in Section §5.3 for the definition of PPIs considering variability in two ways: the first one is shown with the graphical representation of the PPINOT metamodel, and the second one is extending the formal definition previously presented in Section §2.6.3.

5.4.1 Variability in the Extension of the PPINOT Metamodel

The extension of the PPINOT metamodel is depicted in Figures §5.3, §5.4 and §5.5 as UML diagrams. White boxes represent original PPINOT classes and white boxes with the name underlined represent business process elements with which the PPINOT elements are related. Gray boxes represent new classes required to incorporate variability concepts in the metamodel and PPINOT classes modified to be adapted to variability concepts.

Figure §5.3 is the core of the metamodel, in which the relationship between a PPI definition and a process family is expressed. This relationship reflects the dimension of change *Dim-1*. A `ProcessFamily` represents a business process expressed as a collection of several processes with similarities and differences among them. Each of these members is a `ProcessVariant`. A `ConditionVariant` class is used for the specification of conditions and restrictions required for the modelling of variability, reason why this class is associated with several classes in the proposal. The `PPIDefinition` class is the main element of the metamodel, which includes general attributes to identify a PPI (e.g. identifier and name) and goal to which the PPI is related. In the PPINOT extension, a `PPIDefinition` can be instantiated in two ways: the first, a *PPI*, which represents traditional PPIs associated with a single business process (a `ProcessVariant`) and also represents a *PPI* defined in the same way for all PVs related to it. The second alternative of instantiation is the `PPIVariantPoint` that indicates a PPI defined in more than one way for a process family, gathering a set of `PPIVariants`. A `PPIVariant` represents each different configuration of a PPI for a set of PVs.

Variability is also reflected at the PPI attributes. For the sake of simplicity and readability, only three of five attributes are included in Figure §5.3: `ResponsibleDefinition`, `InformedDefinition` and `MeasureDefinition`. `ProcessInstanceFilter` and `Target` are described in Figures §5.4 and §5.5 respectively. Original `Responsible` class was replaced by the `ResponsibleDefinition` class, which can be instantiated by means of `Responsible` and `ResponsibleVariationPoint`. `Responsible` remains as the option to represent a human resource responsible for the PPI regardless of variability. `ResponsibleVariationPoint` indicates that this attribute varies. `Responsibl-`

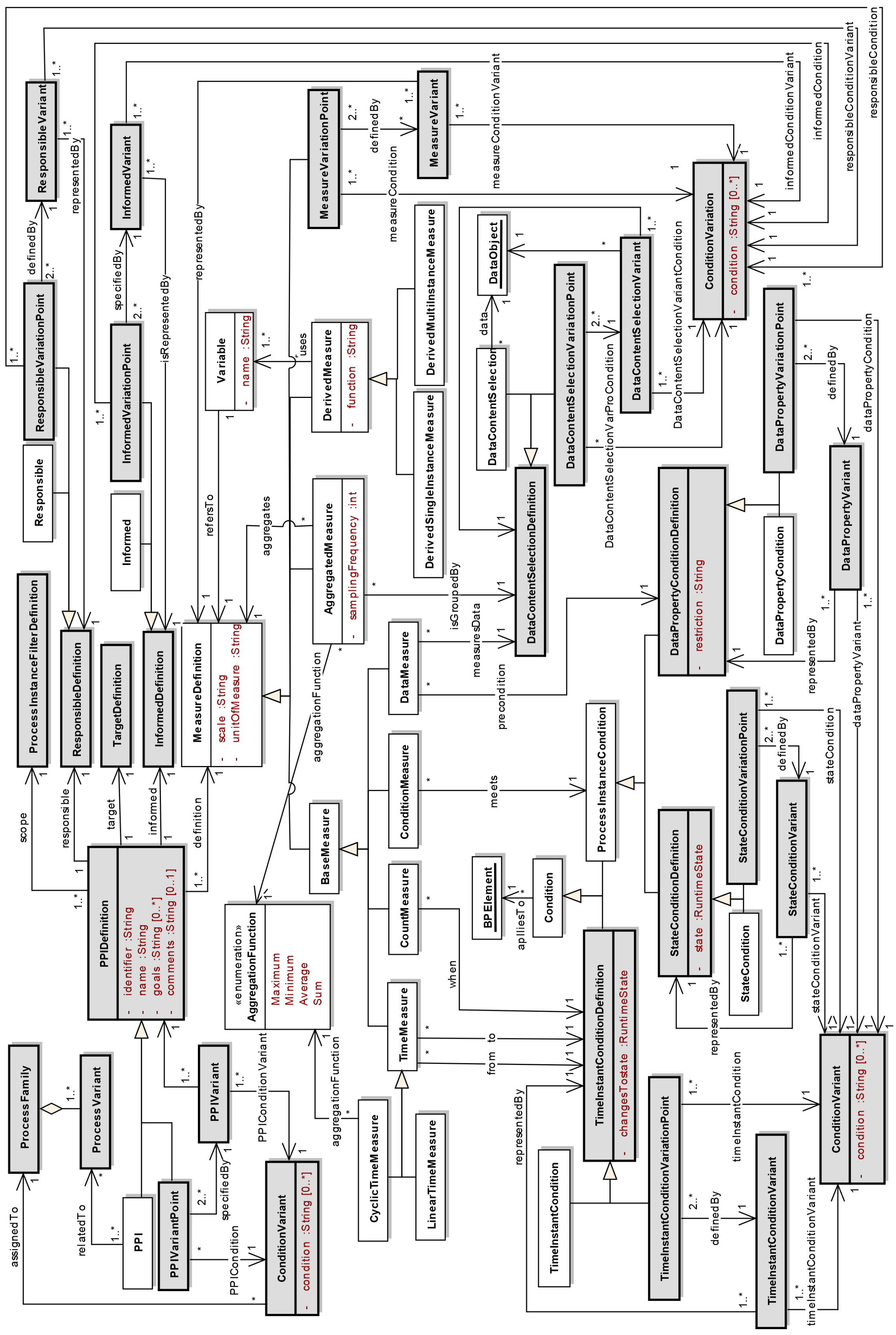


Figure 5.3: Extension of the PPINOT Metamodel considering variability

eVariant is a ResponsibleDefinition that represents a specific way in which a ResponsibleVariationPoint may be configured. As we mentioned before, ConditionVariant represents variability conditions. The structure of variability classes used for the ResponsibleDefinition attribute is applied for each class of the metamodel that is extended to support variability in its definition: InformedDefinition, TimeInstantConditionDefinition, StateConditionDefinition and DataContentSelectionDefinition.

MeasureDefinition is a complex attribute that can be instantiated as: BaseMeasure(TimeMeasure, CountMeasure, ConditionMeasure and DataMeasure), AggregatedMeasure and DerivedMeasure. A BaseMeasure is related to business process elements (BPElement) by means of a Condition. Depending on the type of the measure, the condition can be a TimeInstantCondition or a ProcessInstanceCondition. The first one determines when a CountMeasure should count values or the period of time (from-to) to be measured by a TimeMeasure. The second one, a ProcessInstanceCondition evaluates two conditions: A particular state (StateCondition) or a combination of state and data property (DataPropertyCondition). This three types of conditions have been extended to support variability in its definition according to the structure described for the ResponsibleDefinition, where main classes TimeInstantCondition, StateCondition and DataPropertyCondition are replaced by TimeInstantConditionDefinition, StateConditionDefinition and DataPropertyConditionDefinition respectively, and by their respectively related variability classes.

An AggregatedMeasure is defined by aggregating one BaseMeasure or DerivedMeasure using an AggregatedFunction, and their values can be grouped by information taken from a DataObject by means of a DataContentSelection. The structure of variability described for the ResponsibleDefinition class is also applied for the DataContentSelection class. DerivedMeasure has not been extended for variability because only the function attribute defined may change. If this function attribute must change, we consider that the complete measure must change, and for that situation a new measure (MeasureVariationPoint) has been included in the metamodel. A MeasureVariationPoint, similar to the structure of variability expressed before, is introduced as a new way of instantiating a MeasureDefinition (in addition to BaseMeasure, AggregatedMeasure and DerivedMeasure). It is composed by more than one MeasureVariant, representing all those alternatives to define the same measure, but that require some changes in the number or type of its components. Finally, MeasureVariant represents each option of configuration in which a measure can be represented.

The two remaining PPI attributes, Scope and Target, are complex structures, that incorporate variability classes in a similar way as other PPI attributes as follows. The scope, represented by the ProcessInstanceFilter class, provides four alternatives to define a Scope: as a LastInstancesFilter to consider last instances executed, TimeFilter as a temporal condition over process instances, ProcessStateFilter which depends on the state of the process instance and a ComposedFilter that allows combi-

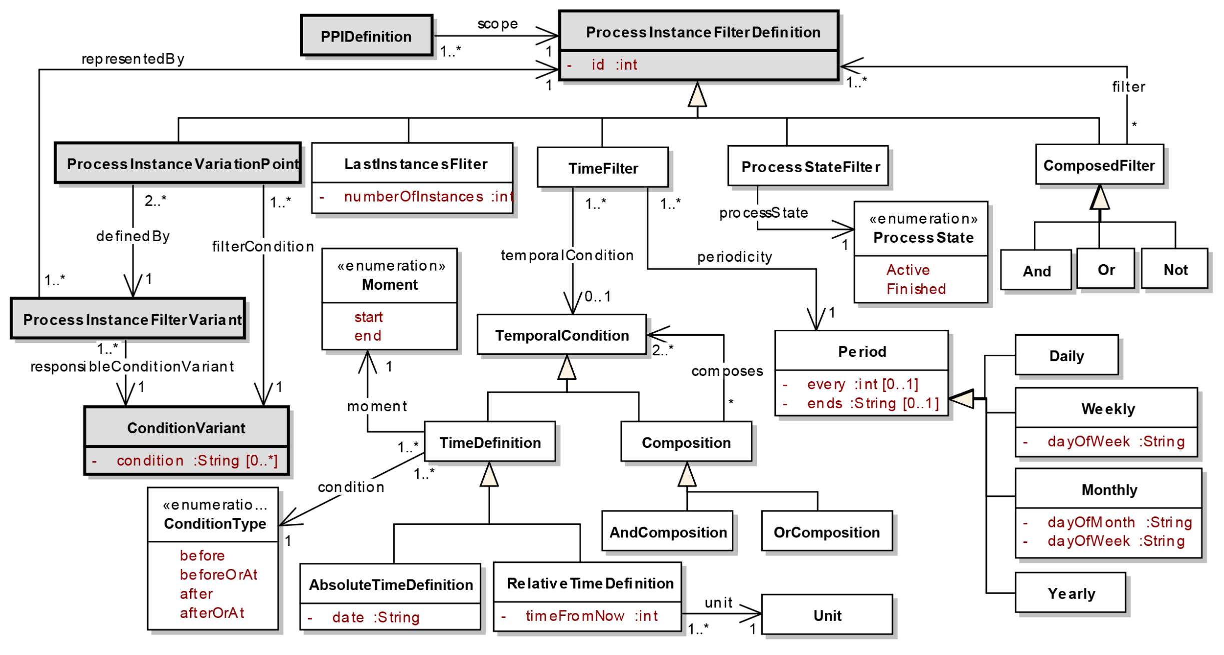


Figure 5.4: Excerpt of PPINOT Metamodel - Scope attribute (*ProcessInstanceFilterDefinition* class)

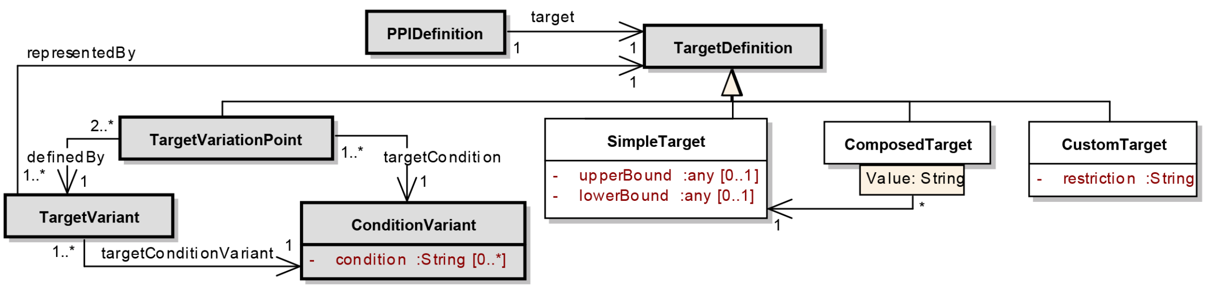


Figure 5.5: Excerpt of the PPINOT Metamodel - Target attribute (*TargetDefinition* class)

nations using logical expressions. Those alternatives represent a filter without variability. `ProcessInstanceVariationPoint` class has been included as a new way to represent points of variability of the filter and `ProcessInstanceFilterVariant` represents a particular variant of the attribute. Similar changes are applied to Target attribute in Figure §5.5, where `SimpleTarget`, `ComposedTarget` and `CustomTarget` are three types of target without variability and `TargetVariationPoint` class represents variability in that attribute.

5.4.2 Variability in the Extension of the PPINOT Formal Definition

In this section, we extend the PPINOT metamodel using the formal definition previously introduced in Section §2.6.3. In this extension we specify the relationship between original PPINOT concepts and the concepts of variability as we described in the previous section.

The PPINOT performance model cannot model the variability identified in Section §5.3. To solve it, we introduce a *variable performance model* as an extension of a PPINOT performance model PM where PPIs, measures and connectors for linking measuring elements with business process elements or amongst them vary depending on the process variant to which they are applied. However, we need first to formally define what we understand as a process family and process variant.

Definition 5.1 - Process family.
A process family $PF = \{pv_1, pv_2, \ldots, pv_n\} = \{bp_1, bp_2, \ldots, bp_n\}$ is a set of business processes that share some common elements. Each $bp_i \in PF$ is called a process variant (pv).

This definition do not intend to be complete, but it just focuses on the elements that are relevant for variable performance models.

With this definition of process family, a variable performance model can be defined as follows.

Definition 5.2 - Variable performance model.
Let $PF = \{bp_1, \ldots, bp_n\}$ be a process family, $\mathcal{PF} = \mathcal{P}(PF) \setminus \varnothing$ be the power set of PF without the empty set, and $\mathcal{C}_{PF} = \mathcal{C}_{bp_1} \cup \ldots \cup \mathcal{C}_{bp_n}$ be the set of possible conditions defined over any process in the process family, a variable performance model is a tuple $PM^V = (P, M, L_P, L_M, P^V, L_P^V, L_M^V)$, where:

- P, M, L_P, L_M refer to elements of a performance model defined over $\mathcal{C}_{PF}$.
- $P^V : P \to \mathcal{PF}$ defines the process variants to which each PPI applies.
- $L_P^V : L_P \to \mathcal{PF}$ defines the process variants to which each link between a PPI and its attributes applies.
- $L_M^V : L_M \to \mathcal{PF}$ defines the process variants to which each link between measures or between a measure and a process element applies.

Functions P^V, L_P^V and L_M^V introduce the modelling of the variability dimensions described in Section {sec:VariabilityInPPIsDefinition} as follows:

- P^V allows expressing **Dim-1** by providing a mechanism to specify which are the process variants to which a PPI applies.
- L_P^V allows expressing **Dim-2** by providing a mechanism to specify which are the process variants to which the alternative attributes for a PPI apply. This includes target, scope, human resources and measure definition, which are the links included in L_P

- L_M^V allows expressing **Dim-2.M1** and **Dim-2.M2** by providing a mechanism to specify which are the process variants to which the links between measure definitions and process elements (**Dim-2.M1**) or to which a certain structure of a measure definition (**Dim-2.M2**) apply. The former includes *cond* and *data* links, whereas the latter includes *cyclic*, *agg*, *uses* and *derfun* links.

Note that these variability functions can also be defined intensionally, i.e., by defining properties that all process variants to which a certain model element apply must fulfill (e.g., the presence of a certain activity in the variant).

A function that represents the process variants to which each measure applies (M^V) is not necessary because it can be derived from the variability functions of the PPIs (L_P^V) and measures (L_M^V) linked to it as follows:

$$M^V(m) = \bigcup_{(p_i,m)\in mes} L_P^V(p_i,m) \cup \bigcup_{(m_i,m)\in agg} L_M^V(m_i,m) \cup \bigcup_{(d_i,m,x)\in uses} L_M^V(d_i,m,x)$$

Based on these definitions, the concept of a syntactically correct variable performance model can be defined. In short, a syntactically correct variable performance model adds the necessary requirements to PM^V that ensure that each process variant has a syntactically correct performance model.

Definition 5.3 - Syntactically correct variable performance model.
Let PF be a process family, $\mathcal{PF} = \mathcal{P}(PF) \setminus \varnothing$ the power set of PF, $\mathcal{C}_{PF} = \mathcal{C}_{bp_1} \cup \ldots \cup \mathcal{C}_{bp_n}$ be the set of possible conditions defined over any process in the process family, $PM^V = (P, M, L_P, L_M, P^V, L_P^V, L_M^V)$ is syntactically correct if it fulfils the following requirements:

1. There is at least one PPI for each process variant: $\forall bp_i \in PF(\exists p_i \in P(bp_i \in P^V(p_i))$
2. Each PPI attribute can only have exactly one single value linked to a PPI p in each variant in which the PPI applies $P^V(p)$, except for the informed attribute: $\forall p \in P, t \in T_P \setminus \{inf\}(\bigcup_{lp\in L_P[p,t]} L_P^V(lp) = P^V(p) \wedge \forall lp_i, lp_j \in L_P[p,t](lp_i \neq lp_j \Rightarrow L_P^V(lp_i) \cap L_P^V(lp_j) = \varnothing)$
3. Measures have at most one link for each possible type of link in L_M except for *uses* in each variant: $\forall m \in M, t \in T_M \setminus \{uses\}(\forall lm_i, lm_j \in L_M[m,t](lm_i \neq lm_j \Rightarrow L_M^V(lm_i) \cap L_M^V(lm_j) = \varnothing))$
4. Depending on its type, measures require at least one element of their links in each variant:
   - $\forall tm \in TimeM(\bigcup_{lm\in L_M[tm,from]} L_M^V(lm) = M^V(m) \wedge$ $\bigcup_{lm\in L_M[tm,to]} L_M^V(lm) = M^V(m))$
   - $\forall cm \in CountM(\bigcup_{lm\in L_M[cm,when]} L_M^V(lm) = M^V(m))$

- $\forall sm \in StateM(\bigcup_{lm \in L_M[sm,meets]} L_M^V(lm) = M^V(m))$
- $\forall dm \in DataM(\bigcup_{lm \in L_M[dm,data]} L_M^V(lm) = M^V(m))$
- $\forall am \in AggM(\bigcup_{lm \in L_M[am,agg]} L_M^V(lm) = M^V(m))$
- $\forall dm \in DerM(\bigcup_{lm \in L_M[dm,derfun]} L_M^V(lm) = M^V(m))$
- $\forall dm \in DerM(\bigcup_{lm \in L_M[dm,uses]} L_M^V(lm) = M^V(m))$

5. Measures must not be applied to variants that do not contain the elements of the process they are linked to: $\forall (m,c) \in cond(\forall bp_i \in L_M^V(m,c)(c \in \mathcal{C}_{bp_i}))$ and $\forall (m,d,s,a) \in data(\forall bp_i \in L_M^V(m,d,s,a)((d,s) \in \mathcal{C}_{bp_i}))$

6. A derived measure cannot be related in each variant to more than one measure with the same identifier, which means that if they have the same identifier, the intersection of their variants must be empty: $\forall (d,m_i,x) \in uses\ \neg\exists (d,m_j,y) \in uses$ $(x = y \wedge m_i \neq m_j \wedge L_M^V(d,m_i,x) \cap L_M^V(d,m_j,y) \neq \varnothing)$

7. The identifiers used for a derived measure in each variant must be sequential: $\forall (d,m_i,x) \in uses(\forall bp_i \in L_M^V(d,m_i,x)(x \leq \mid \{u \in L_M[d,uses] \mid L_M^V(u) = bp_i\} \mid)$.

8. For all $(d,fn) \in derfun, fn \in F$ must be a function defined over the Cartesian product of the set of all possible values of the set of measures linked to d that apply for each variant bp_i to which (d,fn) applies ($\{m \in M \mid (d,m,x) \in uses \wedge bp_i \in L_M^V(d,m,x)\}$), ordered according to x.

Finally, using these definitions, it is easy to obtain a performance model $PM_i = (P_i, M_i, L_{P_i}, L_{M_i})$ for a specific process variant bp_i. For P_i, L_{P_i} and L_{M_i}, it just includes the elements of the variable performance model that apply to the process variant at hand. For M_i, it includes the measures that are used in the links of L_{M_i}. This can be formalised as follows.

Definition 5.4 - Performance model of a process variant.
Let $PF = \{bp_1, \ldots, bp_n\}$ be a process family, and $PM^V = (P, M, L_P, L_M, P^V, L_P^V, L_M^V)$ be a variable performance model of PF, the performance model of a variant bp_i of the process family is a tuple $PM_i = (P_i, M_i, L_{P_i}, L_{M_i})$, where:

- $P_i = \{p \in P \mid bp_i \in P^V(p)\}$
- $L_{P_i} = \{lp \in L_P \mid bp_i \in L_P^V(lp)\}$
- $L_{M_i} = \{lm \in L_M \mid bp_i \in L_M^V(lm)\}$
- $M_i = \{m \in M \mid \exists lm \in L_{M_i}(\Pi_M(lm) = m)\}$

Definition 5.5 - Derivation of a Process Variant.
Let bp, bp′ be business processes, $\mathcal{R}_{bp}$ be the set of all possible rules that can be applied to bp to modify the set of its business process elements, resources or data that define bp to generate bp′. Then, a process variant pv can be defined as the resulting bp′, such that $bp' = pv = (bp, r_{bp})$, where $r_{bp} \in \mathcal{R}_{bp}$.

The set of rules r_{bp} for generating a process variant depends on the variability management strategy selected. In previous sections it was mentioned that variability can be managed in different ways. If we assume variant-based management as a collection of processes, the variability can be managed by restriction (by removing elements of the *bp*) or by extension (by aggregating elements to the *bp*) to generate the process variant.

5.5 Using PPINOT with Variability Business Processes Modelling Languages

As we introduced in previous sections, there are different alternatives for the management of variability in business processes. Here, we focus on the management of variability by restriction and by extension. The survey of La Rosa et al. [90] analyses several proposals for the management of variability and summarises 23 approaches, 11 main proposals and 12 derived from the former. Within the 11 main proposals, we selected 2 of them, one for the management of variability by restriction and another one for the management of variability by extension. The criterion used to select the two approaches was the number of cites of each proposal; information that is also provided in the survey. C-iEPC with 1313 cites and PROVOP with 577 cites, have been selected as alternatives by restriction and extension respectively. PROVOP also supports variability by restriction.

The aim of this section is to illustrate how PPINOT and its extension to support variability can be used in conjunction with different existing proposal for the managing of variability in business process variability; and thus jointly manage both the variability reflected in the business process and in the performance perspective.

5.5.1 Managing Variability by Extension

PROVOP [63] is a framework for the modelling and managing of collections of PVs. This approach uses *a reference/base model* representing the most common parts of the workflow for all PVs involved. This base model is marked with a set of *adjustment points* indicating where the process should be modified to derive a specific PV. The process will be modified with a set of *options* that indicates sequences of change to be applied. Each sequence of changes can be conformed by a set of operations, where each single operation is described.

Figure §5.6 (a) represents a base model resulting for the intersection of the 3 Deliver SCOR process variants. PROVOP is not restricted to a specific business process modelling language. In this example, we use BPMN. For more details about the notation used in the graphical example, see the Appendix §C.2.

For this process model we define a performance model $PM = (P, M, L_P, L_M)$ that includes a PPI (RS.3.51) that measures the execution time of activities *Dx.11*, where x corresponds to the number of the variant. As activities *Dx.11* are not defined in the base model, the PPI is defined but it is not connected to the process (links *from* and *to* are not correctly specified). When the process variant is defined, the PPI must be correctly connected. The performance model PM is initially defined as follows and graphically represented in Figure §5.6 (b) using Visual PPINOT [34].

- $\{RS.3.51\} \in P$;
- $\{aggm_1, tm_1\} \subseteq M$, where $aggm_1 \in AggM, tm_1 \in TimeM$;
- $\{$*(RS.3.51, "all instances"), (RS.3.51, "less than 5 hours"), (RS.3.51, "Responsible Operator 1"), (RS.3.51, "Informed General Manager"), (RS.3.51, aggm*$_1$*)*$\} \subseteq L_P$, where:
  - *(RS.3.51, "all instances")* $\in$ *sco*;
  - *(RS.3.51, "less than 5 hours")* $\in$ *tar*;
  - *(RS.3.51, "Responsible Operator 1")* $\in$ *res*;
  - *(RS.3.51, "Informed General Manager")* $\in$ *inf*;
  - *(RS.3.51, aggm*$_1$*)* $\in$ *mes*;
- $\{(aggm_1, tm_1, AVG), (tm_1, ('', START)), (tm_1, ('', END))\} \subseteq L_M$, where: $(aggm_1, tm_1, AVG) \in agg$; $(tm_1, ('', START)) \subseteq from$; $(tm_1, ('', END)) \subseteq to$;

In PROVOP, a process variant (S') can be derived from a base model (S) as follows $S[\sigma\rangle S'$, where $\sigma =< op_1, op_2, \ldots, op_n >$, and $op_i =< \triangle_1, \triangle_2, \ldots, \triangle_n >$ where $i = \{1, 2, \ldots, n\}$ and $\triangle$ is a process change $\triangle \in \mathcal{C}$ and can be represented as parametrised change operations. A set of process changes are provided by PROVOP for the derivation of a process variant: *INSERT, DELETE, MOVE* and *MODIFY*. In the performance perspective we have a base performance model that needs to be modified. To do that a set of performance configurations needs to be defined. In this case, performance changes (*INSERT, MODIFY* and *DELETE*) can be applied over a PPI, a measure of the PPI or an attribute of a measure. Those performance changes are described bellow:

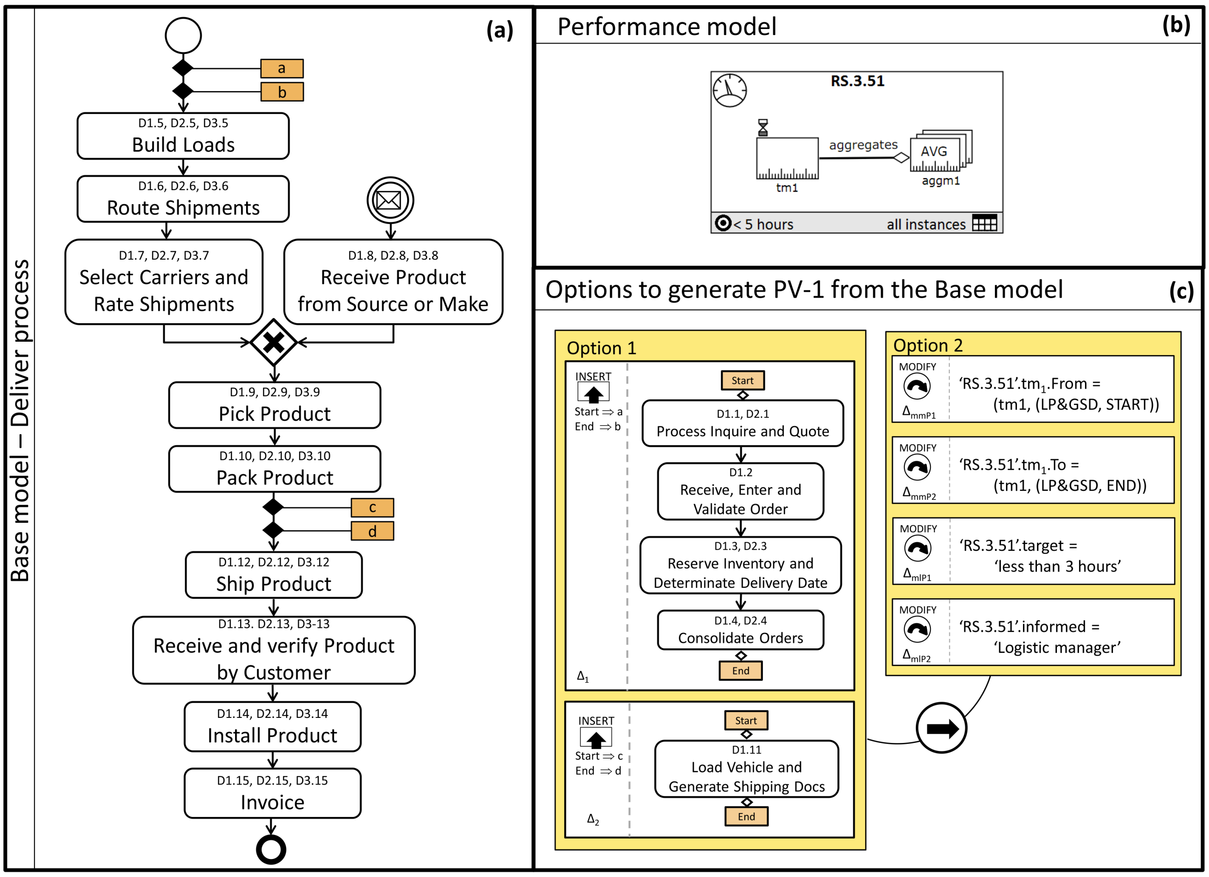


Figure 5.6: (a) Base model in PROVOP for the Deliver SCOR process; (b) a performance mode with a PPI using Visual-PPINOT; (c) PROVOP options to derive a PV-1 in the business process and in the performance model.

- *Insert PPI*: $\triangle_{iP} = (insert, < Pi >, < ppi_id >)$. This instruction adds a PPI (*ppi_id*) in the subset of PPIs (P_1) used for the PM previously defined. The PPI doesn't necessarily have to be well defined at this moment, but when the complete sequence σ is applied, then the *PM* must be syntactically correct.

  Given $PM = (P,M,L_P,L_M)$, where $P_1 = \{ppi_1,\ldots,ppi_n\} \subseteq P$, and $\triangle_{iP}$ = *(insert,* P_1*, 'RS.3.51')*. Then, $\triangle_{iP} \Rightarrow P_1$ results in $\{ppi_1,\ldots,ppi_n,$ $ppi_{n+1}\} \subseteq P$, where $ppi_{n+1} = RS.3.51 \in P$

- *Modify PPI*: $\triangle_{mlP} = (modify, < ppi\_id > . < ppi\_attribute >, \{sco \mid tar \mid res \mid inf \mid mes\})$, where $< ppi\_id >$ indicates the PPI to be modified and $< ppi\_attribute >$ specifies the PPI attribute to be modified. The last parameter indicates that the value of the attribute to be assigned should be described as a scope, target, human resource or measure, because this instruction (*Modify PPI*) modifies the definition of L_P in the *PM*.

  Given the $PM = (P,M,L_P,L_M)$ defined at the beginning of this subsection, and the sequence of changes $\sigma =< \triangle_{mlP1}, \triangle_{mlP2} >$, σ modifies L_P as follows:

  - $\triangle_{mlP1}$ = *(modify, 'RS.3.51'.target, 'less than 3 hours')*
  - $\triangle_{mlP2}$ = *(modify, 'RS.3.51'.informed, {'Logistic manager', 'General Manager of Production'})*

  Then resulting in {*(RS.3.51, 'all instances'), (RS.3.51, 'less than 3 hours'), (RS.3.51, 'Responsible Operator 1'),(RS.3.51, 'Informed Logistic Manager'), (RS.3.51, 'Informed General Manager of Production'), (RS.3.51,* $aggm_1$*)*} $\subseteq L_P$

- *Modify PPI*: $\triangle_{mmP} = (modify, < ppi\_id > . < measure > . < link\_type >, \{cond \mid data \mid agg \mid uses \mid derfun\})$, where $< ppi\_id >$ indicates the PPI to be modified; $< measure >$ indicates the measure of the PPI to be modified; and $< link\_type >$ specifies the link between the measure and the condition to obtain information from the process.The last parameter of the instruction allows the use of all links included in L_M. The use of this instruction *modify* the definition of L_M in the *PM*.

  Given the $PM = (P,M,L_P,L_M)$ defined at the beginning of this subsection, and the sequence of changes $\sigma =< \triangle_{mmP1}, \triangle_{mmP2} >$, σ modifies L_M as follows:

  - $\triangle_{mP3}$ = *(modify, 'RS.3.51'.*tm_1*.from, (*tm_1*, (LP&GSD, START)))*
  - $\triangle_{mP4}$ = *(modify, 'RS.3.51'.*tm_1*.to, (*tm_1*, (LP&GSD, END)))*

  Then resulting in {*(agg,tm, AVG), (RS.3.51,(tm,(LP&GSD, START))), (RS.3.51,(tm, (LP&GSD,END)))* } $\subseteq L_M$

- *Delete PPI*: $\triangle_{dP} = (delete, < Pi >, < ppi_id >)$. This instruction deletes a PPI (*ppi_id*) in the subset of PPIs (P_1) used for a PM previously defined. This instruction eliminates all references related to the *ppi_id* given, including links between the measure of the PPI and the business process elements.

Given the $PM = (P,M,L_P,L_M)$, where $P_1 = \{ppi_1,\ldots,ppi_n,ppi_{n+1}\} \subseteq P$ and $ppi_{n+1} = RS.3.51$; $\triangle_{dP} = (delete,\ P_1,\ 'RS.3.51')$.
Then, $\triangle_{dP} \Rightarrow P_1$ results in $P_1 = \{ppi_1,\ldots,ppi_n\} \subseteq P$.

- *Insert measure*: $\triangle_{iM} = (insert, < M >, < measure\_type >, < measure\_id >)$. This instruction adds a measure (*measure_id*) of type (*measure_type*) in the set of measures (M) of PM. The measure doesn't necessarily have to be well defined at this moment, but when the complete sequence σ is applied, then the PM must be syntactically correct.

  Given the $PM = (P,M,L_P,L_M)$, where $\{aggm, tm\} \subseteq M$; the performance change $\triangle_{iM} = (insert, M, DerM, 'derived\_measure')$.
  Then, $\triangle_{iM} \Rightarrow M$ results in $\{aggm_1, tm_1, derived\_measure\} \subseteq M$

- *Modify measure*: $\triangle_{mM}$ = (*modify*, <*measure_id*> . <*attribute*>, <*attribute_value*>) modifies the value <*attribute_value*> of the attribute of a measure <*measure_id*>. The instruction $\triangle_{mM}$ = *(modify,'derived_measure'.function,'a+b+c')* for example, is used for changing the function defined in a *derived_measure* $\in$ *DerM*.

- *Delete measure*: $\triangle_{dM}$ = (*delete*, <*measure_id*>) eliminates a measure <*measure_id*> from the set of measures M.

  Given the set of measures $\{m_1,\ldots,m_n,m_{n+1}\} \subseteq M$, where $m_{n+1} = derived\_measure$ and the instruction $\triangle_{dM} = (delete, derived\_measure)$. Then, $\triangle_{dM} \Rightarrow M$ results in $\{m_1,\ldots,m_n\} \subseteq M$.

- *Insert attribute*: $\triangle_{iA} = (insert, < measure\_id > . < attribute >, < attribute\_value >)$. Indicates that an attribute is added to a measure $< measure\_id >$ and a value $< attribute\_value >$ for that attribute is assigned.

  Given the instruction $\triangle_{iA} = (insert, 'derived\_measure'.uses, (derived\_measure, aggm_1, a))$, it adds a connection between two measures (a derived measure and an aggregated measure) with the identifier 'a'.

- *Modify attribute*: $\triangle_{mA} = (< measure\_id > . < attribute\_type >, < attribute\_id >, < attribute\_value >)$ modifies the value of an existing attribute.

  Given the $\triangle_{mA} = ('derived\_measure'.uses, 'a', (derived\_measure, tm_1, a))$, it changes the link between a derived measure and an aggregated measure to assign a link between a derived measure and a time measure (tm).

- *Delete attribute*: $\triangle_{dA} = (delete, < measure\_id >, < attribute\_id >)$ eliminates the attribute $< attribute\_id >$ from the measure $< measure\_id >$.

  Given the instruction $\triangle_{dM} = (delete, 'derived\_measure', 'a')$ eliminates the attribute 'a' from the measure *derived_measure*.

Continuing with the example proposed in figure §5.6 (a), the sequence of instruction required to derived the PV-1 and variations of the PPI (RS.3.51) for that PV (See Table §C.1), can be grafically defined as is shown in Figure §5.6 (c).

According to that, a process variant (PV-1) can be defined as the sequence of options (σ) applied over the base model (BM); $BM[\sigma\rangle$ *PV-1*;
where $\sigma =< op_1, op_2 >$, $op_1 =< \triangle_1, \triangle_2 >$ defined using traditional PROVOP expressions; and $op_2 =< \triangle_{mmP1}, \triangle_{mmP2}, \triangle_{mlP1}, \triangle_{mlP2} >$, where:

- $\triangle_{mmP1}$ = *(modify, 'RS.3.51'.tm*$_1$*.from, (tm*$_1$*, (LP&GSD, START)));*
- $\triangle_{mmP2}$ = *(modify, 'RS.3.51'.tm*$_1$*.to, (tm*$_1$*, (LP&GSD, END)));*
- $\triangle_{mlP1}$ = *(modify, 'RS.3.51'.target, 'less than 3 hours');*
- $\triangle_{mlP2}$ = *(modify, 'RS.3.51'.informed, 'Logistic manager');*

5.5.2 Managing Variability by Restriction

To materialise our proposal based on the managing of variability by restriction, we used the *Configurable Integrated Event-Driven Process Change* (C-iEPC) [88] approach. Figure §5.7 represents a C-iEPC model as the result of three Deliver process variants. In this approach a C-iEPC captures multiple variants of a business process in a consolidated manner, called *configurable integrated process model*, hereinafter *configurable process* for short.

In Figure §5.7, the configurable process is defined by means of nodes (functions/activities, events and gateways) and configurable nodes as a C-EPC [143], but we could also include resources and objects assigned to activities to define a complete C-iEPC. Since in a C-iEPC all alternatives should be represented, we extend this definition to the performance perspective of the process to define a PPI that can be configured following similar rules to derive a variant from the C-iEPC (an iEPC). For more details about the notation used in the graphical example, see the Appendix §C.3.

In our example, a PPI (*RS*.3.51) is related to the three process variants, but in *PV-1* the PPI is related to an activity (*'D1.11 - Load Vehicle and Generate Shipping Documentation (LV&GSD)'*) that does not appear in *PV-2* or *PV-3*. For the *PV-2* and *PV-3* the same PPI is defined over a different activity (*'D2.11, D3.11 - Load Product and Generate Shipping Docs (LP&GSD)'*) and some of its PPI attributes also change. To represent connections between the PPI and the two activities of the different variants, we use a configurable gateway that, depending on the process variant, the link between the measure of the PPI and business process elements changes from one activity to another.

Taking into account the PPI information provided in Table §C.1, the performance model (PM) to represent the *PPI-RS.3.51* related to the three process variants can be represented using the extension of the PPINOT definition as follows:

- PF = { $bp_1, ..., bp_n$ } = {*PV-1, PV-2, PV-3*} is the set of Deliver process variants, where:

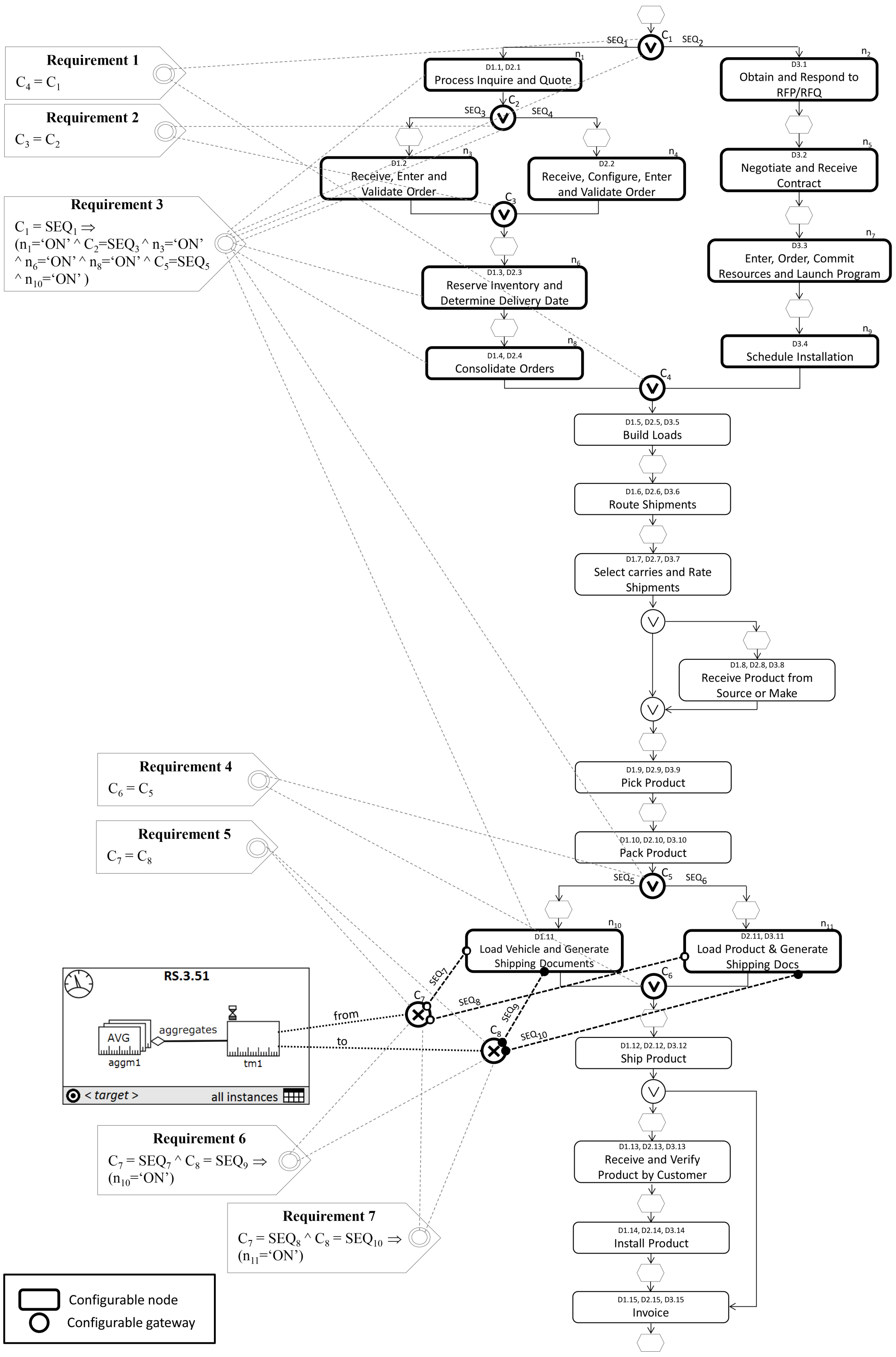


Figure 5.7: A configurable integrated process model (C-iEPC) that represents three process variants of the Deliver SCOR process. To maintain a clear representation of the model, only the requirements necessary to derive *PV-1* have been included.

- *PV-1* : $C_1 = SEC_1 \Rightarrow (n_1 =\text{'ON'} \wedge C_2 = SEC_3 \wedge n_3 =\text{'ON'} \wedge n_6 =\text{'ON'} \wedge n_8 =\text{'ON'} \wedge C_5 = SEC_5 \wedge n_{10} =\text{'ON'})$
- *PV-2* : $C_1 = SEC_1 \Rightarrow (n_1 =\text{'ON'} \wedge C_2 = SEC_4 \wedge n_4 =\text{'ON'} \wedge n_6 =\text{'ON'} \wedge n_8 =\text{'ON'} \wedge C_5 = SEC_6 \wedge n_{11} =\text{'ON'})$
- *PV-3* : $C_1 = SEC_2 \Rightarrow (n_2 =\text{'ON'} \wedge n_5 =\text{'ON'} \wedge n_7 =\text{'ON'} \wedge n_9 =\text{'ON'} \wedge C_5 = SEC_6 \wedge n_{11} =\text{'ON'})$

- $\mathcal{PF} = \mathcal{P}(PF) \setminus \varnothing =$ { {PV-1}, {PV-2}, {PV-3}, {PV-1, PV-2}, {PV-1, PV-3}, {PV-2, PV-3}, {PV-1, PV-2, PV-3} };
- $PM^V = (P, M, L_P, L_M, P^V, L_P^V, L_M^V) =$
  - P, M, L_P, L_M refer to elements of a PM defined over $\mathcal{C}_{PF}$;
    * $RS.3.51 \in P$
    * $\{aggm, tm\} \in M$, where $aggm \in AggM$ and $tm \in TimeM$;
    * $\{(RS.3.51, s_{123}), (RS.3.51, t_1), (RS.3.51, t_{23}), (RS.3.51, hr_{res1}), (RS.3.51, hr_{res23}), (RS.3.51, hr_{inf123}), (RS.3.51, hr_{inf3b}), (RS.3.51, aggm)\} \in L_P$, where:
      · $\{(RS.3.51, s_{123})\} \subseteq sco$, where: $\{s_{123}\} \subseteq \mathcal{S}$;
      · $\{(RS.3.51, t_1, (RS.3.51, t_{23}\} \subseteq tar$, where: $\{t_1, t_{23}\} \subseteq \mathcal{T}$;
      · $\{(RS.3.51, hr_{res1}), (RS.3.51, hr_{res23})\} \subseteq res$, where: $\{hr_{res1}, hr_{res23}\} \subseteq \mathcal{HR}$;
      · $\{(RS.3.51, hr_{inf123a}, (RS.3.51, hr_{inf3b})\} \subseteq inf$, where: $\{hr_{inf123a}, hr_{inf3b}\} \subseteq \mathcal{HR}$;
      · $\{(RS.3.51, aggm)\} \in mes$, where $\{aggm\} \subseteq M$;
    * $\{(tm, (LV\&GSD, START)), (tm, (LP\&GSD, START)), (tm, (LV\&GSD, END)), (tm, (LP\&GSD, END)), (aggm, tm, AVG)\} \in L_M$, where:
      · $\{(tm, (LV\&GSD, START)), (tm, (LP\&GSD, START))\} \subseteq from$;
      · $\{(tm, (LV\&GSD, END)), (tm, (LP\&GSD, END))\} \subseteq to$;
      · $(aggm, tm, AVG) \in agg$, where: $AVG \in \mathcal{F}_{agg}$;
      · $\{(LV\&GSD, START), (LP\&GSD, START), (LV\&GSD, END), (LP\&GSD, END)\} \subseteq \mathcal{C}$;
      · $\{LV\&GSD, LP\&GSD\} \subseteq \mathcal{A}$;
      · $\{START, END\} \in \mathcal{S}_A$;
  - $P^V : P \rightarrow \mathcal{PF} = \{RS.3.51\} \mapsto \{(n_{10} = \text{'ON'}), (n_{11} = \text{'ON'})\}$;
  - $L_P^V = L_P \rightarrow \mathcal{PF} = \{\{(RS.3.51, s_{123}), (RS.3.51, t_{inf123a}), (RS.3.51, aggm)\} \mapsto \{(n_{10} =\text{'ON'}), (n_{11} =\text{'ON'})\} \cup \{(RS.3.51, t_1), (RS.3.51, hr_{res1})\} \mapsto \{(n_{10} =\text{'ON'})\} \cup \{(RS.3.51, t_{23}), (RS.3.51, hr_{res23})\} \mapsto \{(n_{11} =\text{'ON'})\} \cup \{(RS.3.51, hr_{inf3b})\} \mapsto \{(n_2 =\text{'ON'}, n_5 =\text{'ON'}, n_7 =\text{'ON'}, n_9 =\text{'ON'}, n_{11} =\text{'ON'})\}\}$;

- $L_M^V = L_M \rightarrow \mathcal{PF} =$
  $\{\{(aggm, tm, AVG)\} \mapsto \{(n_{10} = 'ON'), (n_{11} = 'ON')\} \cup$
  $\{(tm, (LV\&GSD, START)), (tm, (LV\&GSD, END))\} \mapsto \{(n_{10} = 'ON')\} \cup$
  $\{(tm, (LP\&GSD, START)), (tm, (LP\&GSD, END))\} \mapsto \{(n_{11} = 'ON')\}\};$

C-iEPC derives a process variant from a configurable process model by means of a set of restrictions and guidelines applied to the elements of the process. *Configurable functions* are applied over configurable nodes of the process. A *configurable function* is defined as $f^c \subseteq \{ON, OFF, OPT\}$, whose elements represent that a process element should be included (*ON*), skipped (*OFF*) or conditionally skipped from the model being configurable. For example, the instruction $\{RS.3.51\} \mapsto \{(n_{10}$ = 'ON'), $(n_{11}$ = 'ON')} indicates the PPI is related to the three PVs because in PV-1 the configurable node $(n_{10} = 'ON')$ and for PV-2 and PV-3 the $(n_{11} = 'ON')$ is true. The configurable node n_{11} is related to two process variants, and there is one attribute that is associated with only one of those variants (the PV-3) for that reason, we specify its restriction as a detailed instruction, where all nodes that define the process variant need to be mentioned: $\{(RS.3.51, hr_{inf3b})\} \mapsto \{(n_2 = 'ON', n_5 = 'ON', n_7 = 'ON', n_9 = 'ON', n_{11} = 'ON')\}$.

In order to derive a specific PM for a PV, for example PV-1, from the variable performance model (PM^V), we base on the way a PV is derived from a C-iEPC. Given a configurable process model *cpm*, a process variant *pv* can be derived as $pv = (cpm, \mathcal{C}_{pv})$, where $\mathcal{C}_{pv}$ is the set of configurations, restrictions and guidelines, required to obtain the variant. From the variable performance model PM^V, a variant (pm_{pv}) can be derived as $pm_{pv} = (PM^V, C_{PMpv})$, where C_{PMpv} represents the set of all possible configurations applied to the PM^V to obtain a PM_i and $i = PV1, PV2, PV3$ for our example.

In Figure §5.7, restrictions 5,6, and 7 graphically represent restrictions required to derive the performance model for the process variant PV-1. Using the formal definition of PPINOT and the example included below, the performance model for the process variant PV-1 $pm_{pv1} = (PM^V, C_{PMpv1})$ can be represented as follows:

- $PV\text{-}1: C_1 = SEC_1 \Rightarrow (n_1 = 'ON' \land C_2 = SEC_3 \land n_3 = 'ON' \land n_6 = 'ON' \land n_8 = 'ON' \land C_5 = SEC_5 \land n_{10} = 'ON')$
- $PM_{PV1} = (P, M, L_P, L_M) \mapsto$
  - $RS.3.51 \in P$ and $P \rightarrow \mathcal{PF} = \{RS.3.51\} \mapsto \{(n_{10}$ = 'ON')};
  - $\{aggm, tm\} \in M$, where $aggm \in AggM$ and $tm \in TimeM$;
  - $L_{P(PV1)} \rightarrow \mathcal{PF} =$
    $\{\{(RS.3.51, s_{123}), (RS.3.51, t_1), (RS.3.51, hr_{res1}), (RS.3.51, t_{inf123a}),$
    $(RS.3.51, aggm)\} \mapsto \{(n_{10} = 'ON')\}\};$
  - $L_{M(PV1)} \rightarrow \mathcal{PF} =$
    $\{\{(aggm, tm, AVG), (tm, (LV\&GSD, START)),$
    $(tm, (LV\&GSD, END))\} \mapsto \{(n_{10} = 'ON')\}\};$

The restriction $\{(n_{10} = \text{'}ON\text{'})$ that defines the relation between the PPI and PV-1 is graphically represented in Figure §5.7 by means of the configurable nodes C_7 and C_8, for which SEQ_7 and SEQ_9 (Requirement 6) represent the relationship to PV-1, and SEQ_8 and SEQ_{10} represent the relationship to the configurable node n_{11} and consequently to PV-2 and PV-3 (Requirement 7).

For simplicity, only restrictions related to PV-1 have been included in Figure §5.7.

5.6 Evaluating Variability in Real Scenarios

Our proposal of using PPINOT for modelling PPIs in variability scenarios has been evaluated first, to check that variability in PPIs is a real issue in which there is redundancy of information; second, to verify that dimensions of change proposed in Section §5.3 can be identified in the scenarios analysed; and third, to identify whether there are other examples of variability that may have been omitted in this proposal. Two scenarios have been analysed. For the first one, 120 measures related to two SCOR processes (Deliver and Make) have been modelled. Three process variants of the Deliver process and three of the Make process were considered. For the second scenario, 18 PPIs related to processes for managing incidents in the *Andalusian Health Service (AHS)* were modelled.

Tables §5.1 and §5.2 summarise and classify the set of SCOR measures and PPIs of the AHS. Both tables include: a column indicating the *Dimension* of change; on its right, the ✓ mark and ✗ symbols indicate whether or not there is variability in that dimension. Sub-indexes under symbols indicate sub-dimensions fulfilled in each case. The next column *describes* the dimensions and the last one shows the *number of measures* detected according to the different combinations of variability dimensions found.

For the *SCOR processes*, we modelled and analysed their 120 measures. From them, 32 measures are related to **Dim-1** and 27 are related to **Dim-2**. All measures related to **Dim-2** only represent variability in **Dim-2.M** (*M1, M2*) because there is not information related to other attributes required to define PPIs. As each measure is evaluated according to the two dimensions, table §5.1 presents the six combinations of dimensions identified in this scenario. From it we can say that 42.5% of measures (51 measures) have variability and 57.5% (69 measures) do not have. When there is not variability by **Dim-1**, we found variability by **Dim-2.M1** (8 measures), **Dim-2.M2** (4 measures) and by both sub-dimensions **Dim-2** (7 measures). When there is variability in **Dim-1** we found two combinations: without variability in **Dim-2** (24 measures) and with variability only in sub-dimension **Dim-2.M1** (8 measures) but there are not measures with **Dim-2.M2** or with both of them.

In our second example (the AHS) the variability of other PPI attributes is evidenced. Specifically, values of targets *(T)* or other attributes of the PPI change frequently depending on the priority of the incident (very high, high or normal) handled by the

Table 5.1: Classification of SCOR measures according to dimensions of change.

Dimension		**Description**	**Total**
Dim-1	✗	- Measure is defined in all process variants	69
Dim-2	✗	- Measure is defined in the same way in all process variants	
Dim-1	✗	- Measure is defined in all process variants	8
Dim-2	✓$_1$	- *Dim-2.M1:* The PPI is connected to different BPElements	
Dim-1	✗	- Measure is defined in all process variants	4
Dim-2	✓$_2$	- *Dim-2.M2:* The PPI is calculated using different values	
Dim-1	✗	- Measure is defined in all process variants	7
Dim-2	✓	- *Dim-2.M1, Dim-2.M2:* The PPI is connected to different BP Elements and is calculated using different values	
Dim-1	✓	- Measure is defined in some process variants	8
Dim-2	✓$_1$	- *Dim-2.M1:* The PPI is connected to different BPElements	
Dim-1	✓	- Measure is defined in some process variants	24
Dim-2	✗	- Measure is equal defined in all PV where the PPI appears	
Total of measures			**120**

Table 5.2: Classification of AHS PPIs according to dimensions of change

Dimension		**Description**	**Total**
Dim-1	✗	- Does not vary with regard to the PV	9
Dim-2	✗	- *(Dim-2.T)* Target does not vary	
Dim-1	✗	- Does not vary with regard to the PV	8
Dim-2	✓$_T$	- *(Dim-2.T)* Target varies depending on the priority value	
Dim-1	✗	- Does not vary with regard to the PV	1
Dim-2	✓$_P$	- *(Dim-2.P)* Priority attribute vary	
Total of PPIs			**18**

process. Table §5.2 classifies those PPIs in accordance with our dimensions of change. In this scenario, 50% of measures (9 measures) has variability and the other 50% (9 measures) does not have it. There is not variability according to **Dim-1**, but there is it according to **Dim-2.T** (8 measures) because the target value of the PPI changes. We also found a PPI that changes depending on a priority value. This option is not considered in the definition of a PPI using PPINOT, but can be considered as a new attribute of the PPI, so it is included as variability in **Dim-2** in a sub-dimension **Dim-2.P**.

From both tables we can see that for the two scenarios analysed, variability is present in a high percentage: 42,50% (51 measures) for the case of SCOR and 50% (9 PPIs) for the *Andalusian Health Service*.

5.7 Discussion

Throughout this chapter we have tried to answer the questions posed in Section §5.2 related to variability in business process and the performance measurement of them.

In general, we can conclude that the performance perspective of business processes is subject to variation like other perspectives of business processes. After the analysis of the two scenarios proposed, we identified two main dimensions of change in which variability is reflected in PPI definitions and other sub-dimensions of change related to PPI attributes. These dimensions help to answer VQ1. Some of these dimensions (D2.M1) are related to variations in other perspectives like control-flow, but other dimensions (D2.M2) show that PPIs can also be subject of their own variations regardless of the other perspectives such as changes in the target value of the PPI. Furthermore, the cases that we have analysed show that the variability of PPIs is quite common, affecting almost half of the PPIs defined in each case. All those dimensions of change are not tied to any concrete proposal for modelling PPIs.

Regarding to V-RQ2, we also have identified that it is convenient to develop techniques and models to manage this variability, favour reuse of PPI definitions and reduce design and maintenance time of models. In this sense, we propose to extend the PPINOT metamodel to include concepts of variability and to be able to manage the dimensions of change described in Section §5.3. We chose PPINOT because it provide several characteristics in PPI definitions in tradicional single processes that can be extended to variability scenarios: unambiguous and complete definitions, meet with the *SMART* criteria, has high expressiveness, facilitates traceability with the business process elements §2.6. The extension of the metamodel and the formal definitions presented in Section §5.4 help us to specify restrictions between PPINOT elements and the requirements of a syntactically correct variable performance model that ensure that each process variant has a syntactically correct performance model. In our proof of concept we have shown the possibility of using PPINOT for modelling variability related to the performance perspective in two ways: managing each variant independently of others and managing all process variants as a collection (a process family). By using this second alternative, redundant information derived from information common to all process variants is reduced, the re-use of information is promoted, and if any change needs to be applied to a set of variants, the changes are applied directly to a set of process variants and not to individual process variants, thus reducing possible errors resulting from poorly updated information.

Our proposal is not restricted to a single variability business process modelling language (VBPML). However, due to the fact that a process variant is derived from a performance model of variability by means of a set of configuration instructions, these instructions might be closely linked to the VBPML used, but our proposal supports the necessary modifications to adapt PPINOT to different VBPMLs. Thus answering V-RQ3. Two examples for the managing of variability (one by restriction and one by extension) reflects the flexibility of PPINOT in this way. In the first case, managing variability by restriction, a process is derived following PROVOP directives, being necessary to define a set of operations to define the set of possible sequence of changes to be applied to the base model. In the second example, managing variability by extension using C-iEPC, the derivation of a process variant is more direct because it was not necessary to define additional operations and restrictions to identify a process variant.The derivation is made using configurable functions provided by C-iEPC. Although the proposal can be adapted to different languages, it is true that, given the very different characteristics of each BPML for managing variability, it has not been possible to find a unique way to apply the extension of the metamodel without depending to a large extent on the technique (extension or restriction) used.

Although the graphical representation of the PPINOT extension is not a main objective for this chapter, the first steps have been taken to extend the graphical notation of PPINOT taking into account the VBPML used, as can be seen in figures §5.6 and §5.7. In this sense, like the formal definition, the graphical notation must be modified depending on the VBPML used, which can have a complex learning curve.

To facilitate the modelling of PPIs considering variability concepts, we have proposed the basis for a graphical notation using two different variability languages. However, the main limitation of the graphical proposal is that it is highly dependent on the language used. We have not been able to generate an unified proposal, as the proposals for managing variability by extension and restriction are very different. If we want to integrate the definition of indicators into other proposals, such as those presented in [90], we must adapt the graphical notation to the particular characteristics of that proposal. In fact, these features also affect the application of the metamodel extension, as shown in the examples in Section §5.5.

Since we have only tested the extension of the PPINOT metamodel by means of two VBPMLs, one for each alternative for managing variability (extension and restriction), it would be interesting to apply this extension to other VBPML. By doing this, it would be possible to analyse similarities and/or differences between the different approaches. In this way, it could be concluded whether the application of the extension is entirely dependent of the VBPML or only depends on one of the alternative for managing variability used.

5.8 Summary

In this chapter, we analysed how variability affects the performance perspective of business processes from the definition of PPIs. We proposed a set of dimensions of changes to classify how variability of business processes is reflected in PPI definitions. We extended the PPINOT metamodel with the aim of providing a mechanism for modelling PPIs in processes where several process variants are involved. We showed examples to illustrates that the use of the PPINOT extension helps reduce efforts in the modelling of PPIs and facilitates the management of process variants as a collection and not as individual processes. We also related the extension of PPINOT with two different Variability Business Process Modelling Languages (VBPML) able to manage variability by restriction and by extension. A first version of an extension of the PPINOT graphical notation is also presented in two ways, one for each type of VBPML. Finally, we also evaluated variability with data taken from two different scenarios.

Performance Management of Knowledge-Intensive Processes

"Knowledge has no enemy except an ignorant man."

George Puttenham (1529-1590),
English writer and literary critic. Author of The Arte of English Poesie

KiPs are processes whose execution is heavily dependent on knowledge workers performing various interconnected knowledge-intensive decision making tasks. However, despite the growing body of research focused on understanding KiPs and proposing systems to support them, the research question on how to define performance measures on KiPs is still open. This chapter focuses on addressing this question and is organised as follows: Section §6.1 introduces the chapter. Section §6.2 describes the scenario in which the problem is motivated. Section §6.3 describes an ontology for defining PPIs on KiPs, the KiPPINOT ontology. In Section §6.4, a methodology for the meeting performance goals using the KiPPINOT ontology is proposed, the MPG-K methodology. In Section §6.5, the ontology and the methodology proposed are applied to a real scenario. A brief discussion about both proposals presented in this chapter is included in Section §6.6. Finally, Section §6.7 summarises the chapter.

6.1 Introduction

Knowledge-intensive Processes (KiPs), as discussed in Section §3.3, are processes whose execution is heavily dependent on knowledge workers performing various interconnected knowledge-intensive decision making tasks. Among other characteristics, KiPs are usually non-repeatable, collaboration-oriented, unpredictable and, in many cases, driven by implicit knowledge from participants, derived from their capabilities and previous experiences.

All types of business processes, knowledge-intensive or not, need to be measured, so as to evaluate and continuously improve their performance [101]. However, despite the growing body of research focused on understanding KiPs and proposing systems to support them, the research question on how to define performance measures on KiPs is still open.

Performance management has been widely analysed in the context of structured business processes. Proposals such as [31, 81, 108, 125, 130, 166] provide mechanisms for the definition and monitoring of PPIs in business processes whose expected behaviour is predefined. These proposals provide the opportunity of recognising problems and taking corrective actions before these problems increase, while also facilitating the comparison between an organisation and their competitors [87]. The use of general approaches for the definition of performance measures also reduces the risk of introducing differences in the operationalisation of the performance information, avoiding thus a wrong analysis during decision-making tasks and inconsistencies during information exchange. In addition, these approaches also allow the automation of PPI calculation as well as an easier maintenance, which are time-consuming and error-prone tasks in their absence [34].

However, the aforementioned proposals cannot be used "as-is" in KiPs for one main reason: the different nature of these processes. Those existing proposals have been developed to be applied over structured business processes, with a predefined order in their control flow, and a set of characteristics known a priori for the different business process elements involved. Whereas in KiPs new information arises at runtime, like the explicit knowledge used in process activities or the constraints and rules that have driven actions and decision-making along the process execution. Therefore, existing proposals need to be extended to take all this information into account during the definition of performance measures and to be able to refer to specific knowledge-intensive concepts, like performance measures related to collaboration between process participants, e.g., the measurement of the interaction and messages exchanged among the team members in the context of an IT customer support process.

Some proposals such as [91] and [22] attempt to address the measurement of knowledge-intensive information by proposing concrete metrics for assessing knowledge management performance in specific organisations. However, as far as we know, a general proposal for the definition of performance measures in KiPs that can be used independently of the context and that addresses the issues described above is missing.

Furthermore, in KiPs, there are performance improvements that cannot be hard-wired into the business process model but, rather, must be translated to the participants in the form of performance goals that must be taken into account during process execution. Therefore, a performance management solution for KiPs should help to come up with useful guidelines that enable process participants to meet and improve those performance goals.

In this chapter, we aim at improving the performance management capabilities in KiPs. To do so, we present a twofold contribution. First, we introduce the KiPPINOT ontology, which allows the formal definition of performance indicators in two types of processes: structured and knowledge-intensive. The second contribution is the MPG-K methodology, a methodology for Meeting Performance Goals with KiPPINOT. It relies on KiPPINOT and the concepts of lead and lag indicators [102] in order to provide process participants with actionable guidelines that help them to conduct the KiP in a way that fulfils a set of performance goals. Both the ontology and the methodology have been applied to a case study based on IT incident management processes on real organisation in Brazil.

It should be noted that the main ideas expressed in this chapter have been compiled in a scientific paper that was accepted in September 2018 for its publication under the title “Measuring Performance in Knowledge Intensive Processes”. The citation to this paper is not included in this document because it is not yet available either on the web or on paper. .

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

The main ideas expressed in this chapter have recently been accepted for publication in an article in the ACM-ToIT Journal, and by copyright, they have been omitted from this document.

The name of the paper is: Measuring Performance in Knowledge Intensive Processes.

RELATIONSHIPS BETWEEN DECISIONS AND PERFORMANCE

“It`s not hard to make decisions when you know what your values are.”

Roy Edward Disney (1930-2009),
Businessman

Decision management is of utmost importance for the achievement of strategic and operational goals in any organisational context. Therefore, decisions should be considered as rst-class citizens that need to be modelled, analysed, monitored to track their performance, and redesigned if necessary. In this chapter, we discuss the relationship between decisions and process performance measurement. In Section §7.1, we introduce the chapter. Section §7.2 presents an illustration scenario that will be used along this chapter. Section §7.3 analyses the relationship between decisions and process performance from three points of view. In Section §7.4, those relationships are represented using two tools, DMN and PPINOT. A discussion about this chapter is set out in Section §7.5. Finally, Section §7.6 concludes and summarises the chapter.

7.1 Introduction

Decision management is of utmost importance for the achievement of strategic and operational goals in any organisational context. Therefore, decisions should be considered as rst-class citizens that need to be modelled, analysed, monitored to track their performance, and redesigned if necessary [40]. Here, we focus on decisions made in the context of business processes, and ideally, decisions modelled using DMN or a similar notation. Therefore, these decisions are usually made several times because they are repeated in each process instance.

Up to now, existing literature that studies decisions in the context of business processes has focused on the analysis of the de nition of decisions themselves, in terms of accuracy, certainty, consistency, covering and correctness [72, 131, 132, 133]. However, to the best of our knowledge, no prior work exists that analyses the relationship between decisions and process performance measurement.

In this chapter, we identify and analyse this relationship from three different perspectives. First, we analyse the impact of decisions on process performance and how this information can help the decision-making process. Second, we focus on the performance measurement of decisions themselves based on evidences gathered from the process execution. Finally, we analyse how process performance measures and indicators can be used for the de nition of decisions. Furthermore, we also introduce solutions for the representation of these relationships: the concept of decision performance indicator (DPI) is proposed; PPINOT Metamodel [31] and the DMN standard are extended and integrated to propose a formal alternative for the measurement of decision performance; and the inclusion of performance information in decisions is improved providing more expressiveness by using PPINOT.

It should be noted that the main ideas expressed in this chapter have previously been published in the conference paper [48].

7.2 Problem Illustration

The problem illustration of this chapter is based on Figure §7.1. It shows a DMN model based on real decisions made as part of the IT Incident Management process of a public organisation whose identity or characteristics cannot be revealed for privacy reasons. The model was built with information provided by the organisation and the related data presented along further sections were taken from event logs of its information system. The IT Department receives and records IT incidents in one of its information systems. Incidents are resolved by agents external to the organisation, so before resolving them it is necessary to determine their priority level and the resource responsible for resolving it. By way of example, we focus on a speci c part of the business process, in particular, on decisions related to the priority setting of IT incidents.

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

The main ideas expressed in this chapter were previously published in a conference paper, and by copyright, they have been omitted from this document.

This information can be consulted at:

Estrada-Torres B., del-Río-Ortega A., Resinas M., Ruiz-Cortés A. (2018) On the Relationships Between Decision Management and Performance Measurement. In: Krogstie J., Reijers H. (eds) Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Springer, Cham

DOI: https://doi.org/10.1007/978-3-319-91563-0_19

Part IV

FINAL REMARKS

Conclusions and Future Work

“Everything that has a beginning comes to an end.”

Marcus Fabius Quintilianus (35-100),
Roman rhetorician

This chapter closes this thesis by providing an overview of all the work performed and the contributions presented. In Section §8.1 we outline the content of this manuscript together with the conclusions we have drawn. Section §8.2 summarises results of this thesis that have been presented and published in relevant forums in the scope of BPM research. Finally, Section §8.3 explains limitations identi ed in our approaches and we propose some lines of future work.

8.1 Conclusions

Measuring the performance of business processes allows us to identify whether the strategic goals established by the organisation are being achieved. For this, it is necessary to have mechanisms to de ne measures and indicators in a standard way for different types of business processes. Traditionally, PPIs have been used to measure the ef ciency and effectiveness of business processes. However, although business process management has evolved to allow exibility in the business process control ow, resources or decisions, the techniques for measuring the performance of these new types of business processes or their components have not evolved at the same pace. In this thesis, we understand exibility as a characteristic of business processes that can be re ected in four ways, (i) by means of variability in business processes, from which derives the possibility of (ii) reusing processes and PPI de nitions; (iii) KiPs, which are business processes with characteristics different from processes structured and well de- ned; and (iv) decisions, as relevant elements of business processes. In this sense, this dissertation attempted to enhance the process performance management focusing on the rst stage of the PPI management lifecycle, from the point of view of the modelling perspective of business processes, because, as far as we know, this issue has not been addressed before, which has generated a gap resulting from the evolution of the business processes and their components and the lack of mechanisms to manage process performance taking into account new business process characteristics. In particular, the main goal of this thesis was to provide techniques and mechanisms to improve the current performance management of business processes from a modelling perspective by means of the improvement of PPI de nitions and their management.

In order to achieve this goal, we have presented artefacts that address the management and mainly the de nition of PPIs from four points of view, which are directly related to the four research questions described in the Section §1.2: the rst one is related to the reuse of the de nitions of process performance indicators (RQ-1); the second one (RQ-2) addresses variability in business processes and in PPIs themselves; the third one (RQ-3) is related to the PPI de nitions in Knowledge-intensive Processes; and nally, the fourth question (RQ-4) addresses decisions as a set of business process elements that are becoming increasingly important to be studied separately from business processes, and its relationship with business process performance.

To answer RQ-1, "How can the modelling perspective of process performance management be improved taking into account the reuse of the de nitions of process performance indicators?", we propose the use of abstraction concepts in PPI de nitions. The reuse in PPI de nitions is carried out through the extension of the PPINOT metamodel to include abstraction concepts. This allows, on the one hand, to reduce nal information presented in the business process model, selecting he relevant information that should be included in the business process model with its PPIs, in order to avoid saturating with information that is not relevant to the end user. In addition, the reuse of PPI de nitions in different business processes or performance measures in different

PPI de nitions, facilitate maintenance of PPI de nitions in such a way that future mod-i cations of these de nitions can be applied quicker and more reliable to all processes, since only one de nition, the reusable de nition, must be updated. Information used for this analysis has been obtained from the SCOR reference model.

The second topic addressed was focused on variability, which is related to RQ-2, “How does the variability of business processes affect the modelling of PPIs?”. This question was addressed both the point of view of variability in business process and variability in PPIs themselves. We have identi ed different types of variability and in the two cases studied, we observed that the management of variability reduces the number of PPIs modelled for a family of processes. With this proposal it is possible to de ne PPIs that can be used on a collection of processes (process variants) instead of de ning them individually for each process variant. In this way, if it is necessary to modify a PPI associated with several process variants, changes are applied once on the PPI but they are re ected in all the process variants involved. In addition, we have shown how this proposal of PPI de nitions can be used in an integrated way with different approaches related to variability in business processes and it is not limited to a particular business process modelling language. Given that there are different alternatives for managing variability, for example, by extension using Provop or by restriction using C-iEPC, we adapted our proposal to be used with these two modelling languages. Using the extension of the formal de nition of PPINOT the changes are minimal; when we use the graphical notation of PPINOT, there is greater dependence on the variability business process modelling language used. Information used for this analysis has been obtained from the SCOR reference model and from PPIs related to the IT incident management process of the Andalusian Health Service in Spain.

The third question proposed was focused on KiPs, speci cally RQ-3 is “How can PPIs be modelled in KiPs?” In this sense, we provided two contributions: an ontology named as KiPPINOT as a rst approach for de ning PPIs in KiPs by means of the integration between PPINOT and KIPO (the Knowledge-intensive Process Ontology); and a methodology named as MPG-K (Meeting Performance Goals with KiPPINOT), to guide participants in the implementation of PPIs in accordance to business goals. This methodology can be applied to both structured processes and KiPs, as well as to scenarios for which both kinds of processes can be found. The usefulness of these two contributions, the ontology and the methodology, has been evaluated through their application in a real scenario in a Brazilian ICT outsourcing company, speci cally in the ICT incident troubleshooting process. The insights provided by our approach were considered highly valuable by domain experts of the company, that is already taking them into consideration to implement changes in the process and its execution.

Finally, we addressed RQ-4 “Which are the relationships between decisions and performance of business processes?” In this sense, we identi ed and analysed the relationships between decisions and performance measurement of business processes from three points of view and we also proved that those relationships can be modelled and supported by extending PPINOT, although other similar proposals could also be used.

From that, we can conclude that decision management can be enriched by considering performance indicators and their relationships with decisions. First, by means of the analysis of the impact of decisions in PPIs to identify the more problematic decisions to be analysed and thus preventing an important waste of time and effort that would be put on analysing hundreds of decisions otherwise. Second, by the introduction of the concept of Decision Performance Indicators and the speci c mechanism with templates and patterns provided to model them. And third, by means of the integration of PPIs and measures in de nition of decisions, speci cally, in decision tables modelled using DMN; in this way we involve performance information in de nition of decisions, and decision outcomes can be considered objective data because they are based on real information taken from the process. These relationships have been analysed using real decisions made as part of the IT Incident Management process of a public organisation whose identity or characteristics cannot be revealed for privacy reasons.

To sum up, as a major conclusion of this dissertation, we can state that:

We have demonstrated that it is possible to extend the scope of business process performance measurement in four different ways: by improving traditional mechanisms for de ning PPIs using reuse and abstraction concepts; by taking into account variability in business processes and PPIs; by extending the use of PPIs to Knowledge-intensive Processes; and by analysing the relationship between performance measurement and business process-related decisions. And using PPINOT we have provided mechanisms that allow the modelling of PPIs for the four areas addressed in this thesis.

8.2 Publications

Some of the results shown in this thesis have already been published in scienti c forums or journals, and others, which are also described below, constitute immediate future work because we are nalising details to send them to scienti c journals.

Figure §8.1 summarises and groups these publications taking into account two criteria: type and topic. The four types of publications are related to participations in National Conferences, International Conferences, Doctoral Consortiums and Journals. These types are represented by the background colour of the gure. Regarding the topic criteria, ve topics are depicted in ve different colours: (i) the red line represents publications related to the use of abstraction as a mechanisms for implementing reusein PPI de nitions; (ii) the blue line shows publications about the variability in PPI de nitions; (iii) publications related to the performance management of Knowledge-intensive processes are shown by means of the green line; (iv) publications addressing relationships between decisionsand process performance measurement are shown in yellow; (v) nally, the purple line shows publications that address the problem of improving the performance measurement of business processes as anoverallissue, bring-

Figure 8.1: Publications overview

ing together different proposals addressed in this thesis. Regardless of the type and topic of the publications, their status is indicated by the different kinds of marks explained in the legend of the gure. Some details about these publications are described below.

Reuse-Abstraction

JCIS 2015 [44] B. Estrada-Torres, A. del-Ŕo-Ortega, M. Resinas and A. Ruiz-Cortés. Reduciendo la complejidad grá ca de indicadores de procesos de negocio usando abstracción. In Actas de las XI Jornadas de Ingenieŕa de Ciencia e Ingenieŕa de Servicios (JCIS 2015). Santander, Espãa. September, 2015. [online] Available at: http://hdl.handle.net/11705/JCIS/2015/008

This work proposes the use of abstraction as a mechanism for reusing PPI de nitions and/or parts of them. An extension of the graphical notation of PPINOT is introduced in this proposal.

2018 An extension of the paper presented in the JCIS 2015 conference is about to be submitted to a scienti c journal, including the formal de nition of the PPINOT metamodel, an extension of the PPINOT graphical notation and an extension of the graphical editor as a supporting tool for the modelling of PPIs using reuse of PPI de nitions.

Variability

JCIS 2016 [46] B. Estrada-Torres, V. Torres, A. del-Ŕo-Ortega, M. Resinas, V. Pelechano, A. Ruiz-Cort és. De ning PPIs for Process Variants based on Change Patterns In Actas de las XII Jornadas de Ingenieŕa de Ciencia e Ingenieŕa de Servicios (JCIS

2016). Salamanca, España. September, 2016. [online] Available at: http://hdl.handle.net/11705/JCIS/2016/018

This paper proposes change patterns for de ning PPIs in process families ensuring PPI family correctness de nitions and reducing the number of operations required to specify PPIs.

BPM Forum 2016 [45] B. Estrada-Torres, A. del-Río-Ortega, M. Resinas and A. Ruiz-Cortés. Identifying variability in process performance indicators. In Lecture Notes in Business Information Processing. In: Business Process Management Forum. BPM 2016. Lecture Notes in Business Information Processing, vol 260. Pages 91-107. Springer, Cham. International Conference on Business Process Management, BPM 2016; Rio de Janeiro; Brazil; 18-22 September 2016. DOI: https://doi.org/10.1007/978-3-319-45468-9_6

In this paper, we analyse how variability affects the performance perspective of business processes from the de nition of PPIs. We propose an extension of the PPINOT Metamodel. According to the GII ranking [1] the BPM Conference, where this paper was presented, is classi ed as Class 2, Rating A and Core A

2018 An extension of the paper presented in the BPM Forum is about to be submitted to a scienti c journal. This proposal includes an extension of the PPINOT graphical notation for modelling PPIs in variability scenarios and we explain how this proposal is integrated and used with other Variability Business Process Modelling Languages such as PROVOP or C-iEPC.

KiPs

ACM-ToIT 2016-2018 B. Estrada-Torres, P. H. Piccoli Richetti, A. del-Río-Ortega, F. Araujo Baião, M. Resinas, F. Maria Santoro, and A. Ruiz Cortés. Measuring Performance in Knowledge Intensive Processes

This paper was submitted for a special issue of the ACM-ToIT journal. This journal has a Journal Impact Factor of 1.727 (in 2017) and it is classi ed within quartile 2 (Q2). This paper was accepted in September 2018, so as of the date of submission of this document, the paper is not yet available online or in print.

This article is the result of a research stay at the Federal University of the State of Rio de Janeiro (UNIRIO) in the Department of Applied Informatics. In this paper we propose the KiPPINOT ontology with the aim of providing a mechanism for the de nition of PPIs in Knowledge-intensive processes and the MPG-K methodology that is based on the ontology proposed and on concepts of lead and lag indicators to guide participants in the implementation of PPIs in accordance to business goals. Both artefacts are applied in a case study in a Brazilian company.

JCIS 2017 [47] B. Estrada-Torres, A. del-Río-Ortega, M. Resinas, A. Ruiz-Cortés. On the feasibility of measuring performance using PPINOT in CMMN. In Actas de las

[1]GSS ranking - http://gii-grin-scie-rating.scie.es/

XIII Jornadas de Ingeniería de Ciencia e Ingeniería de Servicios (JCIS 2017). La Laguna (Tenerife), España. Julio, 2017. [online] Available at: http://hdl.handle.net/11705/JCIS/2017/021

The purpose of this paper is to identify and to analyse the feasibility of using an existing mechanism for the de nition and modelling of process performance indicators (PPINOT) in a different context to structured BPMN processes, such as Cases, usually modelled using CMMN.

Decisions

CAiSE 2018 [48] B. Estrada-Torres, A. del-Río-Ortega, M. Resinas, A. Ruiz-Cortés. On the Relationships Between Decision Management and Performance Measurement. In: Advanced Information Systems Engineering. CAiSE 2018. Lecture Notes in Computer Science, vol 10816. Pages 311-326. Springer, Cham. 30th International Conference on Advanced Information Systems Engineering, CAiSE 2018; Tallinn; Estonia; 11-15 June 2018. DOI: https://doi.org/10.1007/978-3-319-91563-0_19.

In this paper we identify and analyse relationship between decisions and business processes performance from three different perspectives, namely: the impact of decisions on process performance, the performance measurement of decisions, and the use of performance indicators in the de nition of decisions; and we also propose solutions for the representation of these relationships based, amongst others, on the DMN standard. Furthermore, we also introduce solutions for the representation of these relationships. According to the GSS ranking [2] the CAiSE Conference, where this paper was presented, is classi ed as Class 2, Rating A and Core A

Overall

SPLC 2017 - Doctoral Symposium B. Estrada-Torres. Improve performance management in exible business processes. In ACM International Conference Proceeding Series Volume 2. Pages 145-149. 21st International Systems and Software Product Line Conference, SPLC 2017; Sevilla; Spain; 25-29 September 2017. DOI: 10.1145/3109729.3109746

This article brings together the main ideas of this thesis. It was presented in the Doctoral Symposium of SPLC Conference 2017 with the aim of obtaining feedback from experts in the eld. This conference is classi ed as Class 2 and Rating A according to the GSS ranking.

BPM 2017 - Doctoral Consortium B. Estrada-Torres. Improving the Performance Management in Business Processes. 15th International Conference on Business Process Management, BPM 2017; Barcelona; Spain; 10-15 September 2017.

[2]GSS ranking - http://gii-grin-scie-rating.scie.es/

This paper gathers the main ideas of this thesis. It was presented in the Doctoral Consortium of BPM Conference 2017 with the aim of obtaining feedback from experts in the eld of Business Process Management.

8.3 LIMITATIONS AND FUTURE WORK

In this section we present the main limitations we have identi ed related to the approaches presented in this thesis and provide some ideas for future work.

Scalability When we addressed process performance measurement taking into account variability in business processes and PPIs, we proposed the extension of the PPINOT graphical notation to facilitate the understanding of reuse of PPIs for the different business process variants. When we integrated our proposal with C-iEPC and the business process modelled has a large number of PPIs, the nal model can be dif cult to understand. We could say that this part of the proposal has limited scalability. However, this limitation is derived from approaches that address the management of variability by restriction, since the process model includes all possible variants for the process, which already from the beginning provides a great deal of information that can be dif cult to understand. The use of PPINOT templates and linguistic patterns for PPI de nitions is a useful alternative to reduce this scalability problem.

Scarcity of data provided by information systems As we mentioned in previous sections, traditional information systems do not typically record information from exible business processes, in fact, some organisations are not able to give value to this information. In several scenarios discussed in this thesis, it has been dif cult to nd information related to, for example, certain aspects of knowledge-intensive processes or decisions, because information systems do not register for example, the information about the instant when a decision starts to be considered or the experience of a decision-maker.

To the extent that information systems are capable of recording this type of information, the management of information and its analysis may be automated to provide more valuable performance information in different scenarios, beyond the traditional structured processes. Nevertheless, the quick pace at which business process management systems are improving their support to decision management and to non-structured business processes makes us think that more and better information concerning them will be available soon, mitigating this issue.

Incomplete tool support During this thesis we have made the rst efforts to extend the PPINOT Tool Suite. We have extended the tool to be able to support abstraction concepts in the de nition of PPIs, allowing thus the reuse of measure de - nitions in different PPIs. However, we have not developed tools for modelling PPIs for all the approaches addressed in this thesis and we have not extended the graphical editor to be able to model PPIs on exible processes or involving decisions modelled with DMN. One of the main reasons is that there is not a tool to model traditional structured processes, knowledge-intensive processes and process families using different business process modelling languages at the same time. Therefore, the implementation of a PPI modelling tool would have to be managed individually in different business process modelling tools and/or build a new tool from scratch. Since PPINOT is not restricted to a particular BPML, one of the clearest future lines of work is oriented to the extension of an existing modelling tool or to the development of a new one, with the aim of modelling PPIs on process families, KiPs or decision-based models. In addition, since PPINOT provides the basis for the automated analysis of PPIs, the implementation of an integrated tool addressing all approaches proposed in this thesis and that allows us to automatically analyse the values obtained from the calculation of PPIs is another clear line of future work.

Part V

APPENDICES

Supply Chain Operation Model Reference

The de nition of business processes is usually derived from the strategic lines and goals of each company. However, some organisations have taken initiatives to propose general recommendations that can be used or adapted to different scenarios. Some of these reference models are the Information Technology Infrastructure Library (ITIL) [8], the Process Classi cation Framework (PCF) [7] and the Supply Chain Operations Reference Model (SCOR) [6], to name a few. These reference models not only focus on the de nition and speci cation of processes, but also propose measures for the evaluation of their performance. According to [41], “Reference models standardize what can be seen as different processes, with unique characteristics and delivering distinguishable products, and how their performance can be measured”.

In this appendix, we focused on describing SCOR, a reference model for evaluating and comparing supply chain activities and performance that are the base of our motivation scenarios in several sections in this dissertation. A supply chain can be de ned as a system of organisations, people, activities, information, and resources that encompass the following three functions: (i) supply of materials to a manufacturer; (ii) the manufacturing process; and, (iii) the distribution of nished goods through a network of distributors and retailers to a nal customer [19, 82].

The Supply Chain Operation Reference model (SCOR) is a process reference model for supply chain management. It enables users to address, improve, and communicate supply chain management practices within and between all interested parties in the enterprise. The SCOR reference model consists of 4 major sections: (i) Performance: Section that provides standard metrics to describe process performance and de ne strategic goals. (ii) Processes: Standard descriptions of management processes and process relationships. (iii) Practices: This section describes management practices that produce signi cant better process performance. (iv) People: Section that provides standard de nitions for skills required to perform supply chain processes.

We focus on two elements of the SCOR structure: processes and performance.

- SCOR processes identify a set of unique activities within a supply chain and describe the business activities associated with all phases of satisfying a demand of customers. These activities are described at a high level of abstraction. SCOR focuses on three process levels and does not attempt to prescribe how a particular organisation should conduct its business or tailor its systems or information ow, because it is out of the scope of SCOR. Figure §A.1 taken from [6], summarises

Figure A.1: SCOR as a hierarchical process model - Figure taken from [6]

and describes the levels of SCOR processes.

- Level-1 processes represent the process types. This level de nes scope and content of the supply chain. There are six Level-1 processes:
  * Plan represents “processes associated with determining requirements and corrective actions to achieve supply chain objectives.”
  * Sourceis described as “The processes associated with ordering, delivery, receipt and transfer of raw material items, subassemblies, product and/or services.”
  * Makeis “the process of adding value to products through mixing, separating, forming, machining, and chemical processes.”
  * Deliver describes “the processes associated with performing customer-facing order management and order ful llment activities.”
  * Return are “The processes associated with moving material from a customer back through the supply chain to address defects in product, ordering, or manufacturing, or to perform upkeep activities.”
  * Enableare “the processes associated with establishing, maintaining and monitoring information, relationships, resources, assets, business rules, compliance and contracts required to operate the supply chain.”
- Level-2 processes are process categories. This level de nes the operation strategy. Each Level-1 process 3 or more differentiating level-2 process cat-

egorisations exist. Plan has 5 level-2 processes: ;Source has 3 level-2 processes: Source Stocked Product, Source Make-to-Order Product and Source Engineer-to-Order Product; Make has 3 level-2 processes: Make-to-Stock, Make-to-Order and Engineer-to-Order; Deliver has 4 level-2 processes: Deliver Stocked Product, Deliver Make-to-Order Product, Deliver Engineer-to-Order Product, Deliver Retail Product; Return has 6 level-2 processes: Source Return Defective Product, Deliver Return Defective Product, Return MRO Product, Deliver Return MRO Product, Source Return Excess Product, Deliver Return Excess Product; Finally, Enable has 9 level-2 processes: Manage Supply Chain Business Rules, Manage Performance, Manage Data and Information, Manage Supply Chain Human Resources, Manage Supply Chain Assets, Manage Supply Chain Contracts, Manage Supply Chain Network, Manage Regulatory Compliance, Manage Supply Chain Risk.

- Level-3 processes describe steps or process elements. This level “de nes the con guration of individual processes”. Here, the focus is on: processes, inputs and outputs, process performance, practices, technology capabilities and skills of staff. For example, the level-2 process “M1-Make-to-Stock” of the level-1 process Make, has the following level-3 elements: M1.1 Schedule Production Activities, M1.2 Issue Material, M1.3 Produce and Test, M1.4 Package, M1.5 Stage Product, M1.6 Release Product to Deliver, M1.7 Waste Disposal.
- Level-4 processes represent activities of implementation of each organisation. This level is out of the scope of SCOR.

Figure §A.2 shows components of the Make process. The three level-2 processes are M1 - Make-to-Stock, M2 - Make-to-Order and M3 - Engineer-to-Order. The level three of each process of level-2 is also detailed. M1.1 represents the rst level-3 element of the rst level-2 process (M1) of the Make process. M2.1 represents the rst level-3 element of the second level-2 process (M2) of the Make process; and so on for all processes.

- SCOR Performance is composed of two elements: Performance Attributes and Performance Metrics. The rst one is composed of a set of performance metrics used to express a strategy. There are 5 types performance attributes: Reliability, Responsiveness, Agility, Costs and Asset Management Ef ciency. The second element measures the ability of a supply chain to achieve these strategic attributes. SCOR metrics (hereinafter SCOR measures) are also distributed in three levels. According to the SCOR speci cation, levels of measures can be described as follows:

  - Level-1 measures are diagnostics for the overall health of the supply chain. These metrics are also known as strategic metrics and key performance indicators (KPI). Benchmarking level-1 metrics helps establishing realistic targets to support the strategic directions. Level- measures are directly related

Figure A.2: Example of level processes of the Make SCOR process

to performance attributes. Perfect Order Ful lment is level-1 measure related to reliability; Order Ful lment Cycle Time is related to the responsiveness attribute; Upside Supply Chain Flexibility, Upside Supply Chain Adaptability, Downside Supply Chain Adaptability and Overall Value at Risk are related to the agility attribute; Total Cost to Serve is related to cost attribute; Cash-To-Cash Cycle Time, Return on Supply Chain Fixed Assets and Return on Working Capital are related to Asset Management.

– Level-2 measures serve as diagnostics for the level-1 metrics. The diagnostic relationship helps to identify the root cause or causes of a performance gap for a level-1 measures. Each level-1 measure can be related to one or more level-2 measures. For example, the level-1 measure Perfect Order Ful lment is calculated as "[Total Perfect Orders] / [Total Number of Orders] x 100%" but to calculate the Total Perfect Orders value information is taken from four level-2 measures: Percentage of Orders Delivered In Full, Delivery Performance to Customer Commit Date, Documentation Accuracy and Perfect Condition. A similar situation occurs with the other level-1 measures.

– Level-3 measures serve as diagnostics for level-2 measures. Similar to level-1 measures, level-2 measures require information taken from level-3 measures. For example, the level-2 measure Percentage of Orders Delivered In Full requires information from two level-2 measures: Delivery Item Accuracy and Delivery Quantity Accuracy; or the level-2 measure Documentation Accuracy that requires information from four level-3 measures: Compliance Documentation Accuracy, Other Required Documentation Accuracy,

Figure A.3: Example of hierarchy in SCOR measures.

Payment Documentation Accuracy and Shipping Documentation Accuracy.

Figure §A.3 shows an example of a SCOR measure of level-1 (Perfect Order Ful l-ment) as a hierarchical representation detailing the level 2 and 3 measures related to it. The level-1 measure is related to four level-2 measures; and each level-2 measure is related to 2, 2, 4 and 5 level-3 measures, respectively.

Examples of Reuse and Abstraction using PPINOT

In this appendix, we apply abstraction concepts described in Chapter §4 using on the one hand, the extension of PPINOT templates and linguistic patterns; and on the other hand the extension of the PPINOT core tool, a Java application for the automatic analysis of PPINOT models.

B.1 PPINOT Templates and Linguistic-Patterns for the Modelling of SCOR Patterns

In this section, we present several MeasureDe nition using linguistic-patterns of PPINOT. Each linguistic-pattern describes one of the patterns listed in Table §4.2. Each patter is represented by means of a CompositeMeasure, speci ed as an ExpandedMeasure. For each de nition, a brief description of measures involved is presented. As other examples presented in Section §4.5, name elements starting with "p" represent parameters to be de ned for the user in the stantiation of a measure.

B.1.1 Sum_Time

This pattern is de ned using an AggregatedMeasure over a Timemeasure that speci es two time instants. The function sum could be a parameter and be de ned by the user, but in this case it is a xed value because represents a type of measure de ned in SCOR.

MeasureDe nition::=
the composite measure Sum_Time is calculated as the SUM of the duration between the time instant when
< p_type$_1$ > < p_name$_1$ > becomes < p_state$_1$ > and
< p_type$_2$ > < p_name$_2$ > becomes < p_state$_2$ > ,
which can be instantiated as: "the measure Sum_Time calculates the total sum of time [< p_description >] between
(< p_type$_1$ > , < p_name$_1$ > , < p_state$_1$ >) and
(< p_type$_2$ > , < p_name$_2$ > , < p_state$_2$ >)

B.1.2 Sum_Cost

This pattern is de ned using an AggregatedMeasure over a DataMeasure that obtains values from a particular DataObject . The function sum is applied over the data object property to be de ned of the DataObject to be speci ed in the instantiation.

MeasureDe nition::=
the composite measure Sum_Cost is calculated as the SUM of the value of property < p_dobject_property> of data object < p_dobject_name>
which can be instantiated as: “the measure Sum_Cost calculates the < p_description> (< p_dobject_property> , < p_dobject_name>)

B.1.3 Sum_Cost_List

This pattern is de ned using an AggregatedMeasure over a ListMeasure . The ListMeasure is formed by 'n' that user should specify in each instantiation. The ListMeasure sums 'n' values in one business process instance. The AggregatedMeasure sums values calculated for the list in each business process instance.

MeasureDe nition::=
the composite measure Sum_Cost_List is calculated as the SUM of a function list SUM over < n > data measures de ned as f the value property < p_dobject_property$_i$ > of data object < p_dobject_name$_i$ > g$_{= 1,...,n}$

B.1.4 Average_Time

This pattern is de ned using an AggregatedMeasure over a TimeMeasure that speci es two time instants. The function average is used in this pattern. Patterns Average_Time and Sum_Time are very similar and could be de ned as the same pattern if the function is expressed as a parameter. In that case, a generic name should be assigned to the pattern.

CompositeMeasure::=
the composite measure Average_Time is calculated as the AVERAGE of the duration between the time instant when
< p_type$_1$ > < p_name$_1$ > becomes < p_state$_1$ > and
< p_type$_2$ > < p_name$_2$ > becomes < p_state$_2$ > ,
which can be instantiated as: “the measure Average_Time calculates the total average of time [< p_description>] between (< p_type$_1$ > , < p_name$_1$ > , < p_state$_1$ >) and (< p_type$_2$ > , < p_name$_2$ > , < p_state$_2$ >)

B.1.5 Average_Time_List

This pattern is defined using an AggregatedMeasure that calculates the average of time values provided by the ListMeasure of each instance. Each ListMeasure calculates the average of a set of time values. When the Average_Time_List is instantiated, the number of elements conforming the list is the same for all process instances.

MeasureDefinition::=

the composite measure Average_Time_List is calculated as the AVERAGE of a function list AVERAGE over < n > time measures defined as { the duration between the time instant when < p_type$_{1i}$ >< p_name$_{1i}$ > becomes < p_state$_{1i}$ > and < p_type$_{2i}$ >< p_name$_{2i}$ > becomes < p_state$_{2i}$ > }$_{i=1,...,n}$

B.1.6 Percentage_Data

This pattern is defined using a DerivedMeasure in which a percentage function is expressed using two values. The first value, partial, is calculated using an Aggregated-Measure that sums values from a DataMeasure that takes its value from a DataObject. The second value, total, is also calculated using an AggregatedMeasure over a Count-Measure. The percentage can be calculated using other sources to take values, but in this case, because of problems found, the value is taken from a DataObject.

CompositeMeasure::=

the composite measure Percentage_Data is calculated as the function partial / total $*$ 100 where

partial is the SUM of the value property < p_dobjectproperty$_1$ > of data object < p_dobjectname$_1$ >,

total is the SUM of the number of times data object < p_dobjectproperty$_2$ > becomes < p_dobjectname$_2$ >.

which can be instantiated as: “the measure Percentage_Data calculates the percentage of

< p_description$_1$ > (< p_dobjectname$_1$ >,< p_dobjectproperty$_1$ >) from < p_description$_2$ > (< p_dobjectname$_2$ >,< p_dpbjectproperty$_2$ >)”.

B.1.7 Complex_Percentage

This pattern is defined for the purpose of including more than one condition in the calculation of the percentage. Similar to occurs in Percentage_Data, Complex_Percentage is calculated taking into account data object properties, but in this case more than one property is considered to the same DataObject. A DerivedMeasure is defined with a function that involves two values. Partial is defined using an AggregatedMeasure that sums values provided by a ListMeasure. By means of a logic function (IsTrue), the List-Measure evaluates if all elements in the list fulfil the requirement specified by the data object properties. If one of them does not fulfil the property, the value is not including in the sum made by the AggregatedMeasure. The second value of the function is total, which is calculated using and AggregatedMeasure that sums values provided by a CountMeasure over a DataObject .

CompositeMeasure =
 the composite measure Complex_Percentage is calculated as the function partial/total $*$ 100 where
  partial is the SUM of a function list IsTrue over < n > data measures, de ned as f the value of property < p_dobject.property$_i$ > of data object < p_dobject.name > g$_{i=1,...,n}$,
  total is the SUM of the number of times data object < p_dobject.name > becomes < p_dobject.property >

B.1.8 Number

This pattern is de ned using an AggregatedMeasure that sums values obtained over a CountMeasure over a DataObject. A counting can be applied over different business process elements, but in our scenario only counting over data objects were found.

CompositeMeasure =
 the composite measure Number is calculated as the SUM of the number of times data object < p_name > becomes < p_state >

B.1.9 Function_5

This pattern is base on the `5 point rolling average' formula used in SCOR, which uses a combination of both historical (4 values) and forward-looking data (one value) to calculate the average. This value is divided by `a total value' of different costs divided by 365. To calculate this formula a DerivedMeasure is de ned to specify the general formula that involves two variable (5points / (total/365)). The rst, 5points, is calculated with a DerivedMeasure that captures values from 5 different data measures, which, in turn, obtain their values from a data object. The second value, total, is calculated using a data value that obtain the total cost registered in a particular DataObject .

CompositeMeasure =
 the composite measure Function_5 is calculated as the function 5points / (total / 365), where
  5points is the function (a+b+c+d+e)/5, where
   a is the value < p_dobject.property$_1$ > of data object < p_name$_1$ >,
   b is the value < p_dobject.property$_2$ > of data object < p_name$_2$ >,
   c is the value < p_dobject.property$_3$ > of data object < p_name$_3$ >,
   d is the value < p_dobject.property$_4$ > of data object < p_name$_4$ >,
   e is the value < p_dobject.property$_5$ > of data object < p_name$_5$ >
  total is the value
   < p_dobject.property > of data object < p_dobject.name >

B.2 EXTENDING THE PPINOT TOOL TO SUPPORT REUSE OF PPI DEFINITIONS

PPINOT, as is de ned in its web page [1], "is a set of libraries aimed at facilitating and automating the PPI management. The support includes their de nition using either a graphical or a template-based textual notation, their automated analysis at design-time, and their automated computation based on processing an event log obtained from a process simulator or a Business Process Management System".

Our rst efforts to incorporate the concepts of abstraction into the PPINOT tools were focused on extending the Visual PPINOT graphic editor. The graphical editor provides a toolbar with general actions (copy, cut, paste, etc.); a working area where business processes and indicators are modelled; and a panel of graphical elements called the "Shape Repository" that contains elements of the graphical notations of BPMN 2.0 and PPINOT. Among the elements of PPINOT we added seven categories to represent graphical elements presented in Figure §4.6.

- Composite Measures, where we included CollapsedMeasure and ExpandedMeasure.
- Base Connector. In it we include the time, count, state condition and data connectors to connect a CollapsedMeasure with its corresponding BPElement (e.g. activity, event, data object, etc.).
- Aggregated Connector. In it we include the time, count, state condition and data aggregated connectors to connect a CollapsedMeasure with its corresponding BPElement (e.g. activity, event, data object, etc.).
- Base List Connectors. In it we include the time, count, state condition and data connectors to connect a CollapsedMeasure with its corresponding BaseMeasure.
- Aggregated List Connectors. In it we include the time, count, state condition and data aggregated connectors to connect a CollapsedMeasure with its corresponding AggregatedMeasure.
- Parameter Connectors. This category includes two elements. The rst one is the Base Parameter Connector that is used for the connection between a CollapsedMeasure with an ExternalValue. The second one is the List Parameter Connector that is used for the connection between a CollapsedMeasure with a list of ExternalValue.
- List Measures. This category contains the two types of ListMeasure. A BaseListMeasure that expects to receive certain information from a set of BaseMeasaure

[1] http://www.isa.us.es/ppinot/

and the Aggregated ListMeasure that expects to receive information from a set of AggregatedMeasaure.

As a second extension of the tool, we focus on the core of PPINOT. The PPINOT tool was developed as a set of Java-Projects that contains and groups de nitions and restrictions speci ed in the PPINOT Metamodel (See Figure §2.5) to manage the modelling of PPIs using its two notations, PPINOT templates and Visual PPINOT as a graphical notation. In this extension, a set of classes and methods were modi ed or implemented to include abstraction concepts and all restrictions required to ensure their proper definition in a model.

All changes were applied in the ppinot-model project, where each PPINOT element is de ned as a Java class. Restrictions and conditions between measures and between measures and business process elements are validated by means of methods de ned in each class. For example, a BaseMeasure de nition is valid (valid()) if it can be identi ed by means of a name different from null or empty, and if all its conditions are associated with a business process element. To describe the PPINOT extension, changes and additions were classi ed in three areas.

B.2.1 Changes in Derived Measures

DerivedMeasure class is modi ed to consider in its de nition the use of ExternalValue. External values are values which are not taken from the process, but they are required to calculate a measure. A DerivedMeasure is valid if it does not have a null or empty name and if it has at least one measure (BaseMeasure or AggregatedMeasure) or ExternalValue connected with it. Its validation is made by means of the valid() method, which returns true if a DerivedMeasure satis es all conditions, otherwise returns false. Measures and external values can be used in the same DerivedMeasure but all of them should be valid elements. An ExternalValue is valid if its name and value are different from null or empty.

ListMeasure class is de ned as a new type of DerivedMeasure, from which it inherits all its attributes (identi er, function, etc.). Unlike DerivedMeasure, a ListMeasure is valid if all its list elements has the same type: all of them are external values, or all of them are measures. If a ListMeasure contains measures, all those measures must have the same type (TimeMeasure, CountMeasure, etc.). Other restrictions about name and number of elements are the same as for the DerivedMeasure.

B.2.2 Connectors as new PPINOT elements

Connector class represents the connection between a CollapsedMeasure and business process elements or between a CollapsedMeasure and other measures. Each Connector used in a CollapsedMeasure represents a measure or external value) in an ExpandedMeasure, consequently the type of connector must coincide with the type of the measure that represents. The complete list of connectors included in the PPINOT extension is presented below:

- CompositeConnector is the general class that represents a connector. It has three attributes: id, name and description. A CompositeConnector is valid if its id and name are different from null or empty.
- MeasureConnector represents the connection between a CollapsedMeasure and a business process element or between a CollapsedMeasure and another measure. Consistent with the types of measures of the PPINOT metamodel, this connector can be used as one of three types: BaseConnector, AggregatedConnector and DerivedConnector.
- ParameterConnector represents the connection between a CollapsedMeasure and an external value provided.
- BaseConnector can be one of four types: TimeConnector, CountConnector, StateConditionConnector and DataConnector. A BaseConnector is valid if its connections, regardless of its type, are different from null and are applied to a business process element. Each BaseConnector inherits attributes from the MeasureConnector class.
- TimeConnector includes two TimeInstantCondition attributes named: ccFrom and ccTo, to indicate the starting and ending point to calculate the measure.
- CountConnector includes a TimeInstantCondition attribute named ccWhen, to indicate the time instant when something happens and should be measured.
- DataConnector includes a DataContentSelection attribute named ccDataContentSelection, to indicate the part of the data object to be measure.
- StateConditionConnector includes a StateCondition attribute named ccCondition, to indicate the ful lment of a state that should be measured.
- DerivedConnector represents the connection between a CollapsedMeasure and a set of measures or external values that will be used in a derived measure, specifically a ListMeasure. A DerivedConnector is valid if attributes like id and name are not null or empty, and if it has at least one element connected with it.
- ListConnector inherits attributes from DerivedConnector. This connector is valid if it has at least one measure or ExternalValue and if it is a valid DerivedConnector.
- AggregatedConnector inherits attributes from MeasureConnector and represents the aggregation of a value over a BaseMeasure An AggregatedConnector is valid if it has a measure to aggregate and that measure is not null.

B.2.3 Composite Measures as new PPINOT elements

CompositeMeasure class is included as a new type of MeasureDefinition , reason why it inherits its attributes id, name, description, scale and unit of measure. As shown in the PPINOT metamodel, a CompositeMeasure can be of two types: CollapsedMeasure and ExpandedMeasure

ExpandedMeasure class represents a measure as an attribute MeasureDe nition from which a reusable structure can be de ned to be reuse. An ExpandedMeasure is not connected with business process elements. This measure only contains the de nition and structure of a measure to be reusable. Those measures that are not connected are identi ed in our applications as LeafMeausures. Each time an ExpandedMeasure de nition is reused, it is necessary to make of copy of it by means of the method clone() , which has the responsibility to copy the structure of the measure.

In the following example averageTime represents the ExpandedMeasure that contains the general structure required to calculate the average time. This measure is reuse when adpCycleTime is assigned with a clone of averageTime. Up to this point, adpCycleTime is not connected with the business process.

```
//Method used to clone an expanded measure
ExpandedMeasure adpCycleTime= averageTime.clone();
```

The de nition of an ExpandedMeasure is valid if its name is not null or empty, if its attribute measure is different from null or empty and if this measure is different from an ExpandedMeasure. In addition, an ExpandedMeasure should have at least one LeafMeasure, otherwise the reuse is not possible.

CollapsedMeasures class is a new PPINOT measure that allows us to instantiate the structure de ned in an ExpandedMeasure, but that contains the speci c connections with business process elements or with other measures. In addition to attributes inherited from the CompositeMeasure, it de nes a expandedCompositeName attribute, as a string to link the CollapsedMeasure with a ExpandedMeasure. Another attribute included in the class is a usedConnectorMap as a Map of pair values (String, CompositeConnector) , to de ne the set of connectors to be used.

A CollapsedMeasure is valid if its name and expandedCompositeName are different from null or empty, if usedConnectorMap is not null, if it has at least one connector assigned and if all its connectors are valid. To validate the latter condition, the method wellConnectedAndNamed() was implemented. This method evaluates one by one all its connectors to determine if they are valid. In other words, if they do not have a null or empty name and all its connections are applied to a business process element, measure or external value.

Instantiation of measures: Finally, after the de nition of composite measures, the connection between a CollapsedMeasure and an ExpandedMeasure is required to implement the reuse of PPINOT measures. A connection implies to make a copy of the ExpandedMeasure and add to this copy the set of connections de ned for a CollapsedMeasure. This action is made by means of the connectCollapsedToExpanded() method, which returns a ExpandedMeasure with the structure cloned from the ExpandedMeasure de ned as a pattern and with all connections de ned in its CompositeMeasure. Before to connect both measures, a set of requirements should be ful lled. If one of them is missing, the connection between them, and the reuse of the ExpandedMeasure is not possible. Below, the method connectCollapsedToExpanded() is shown step by step, describing each instruction in it.

1. Method signature. The ExpandedMeasure pattern is the measure that contains the structure to be reused.

```
public ExpandedMeasure connectCollapsedToExpanded(
                                        ExpandedMeasure pattern)
```

2. Create a new measure. The ExpandedMeasure responseMeasure is declare.

```
ExpandedMeasure responseMeasure = new ExpandedMeasure();
```

3. Validating the ExpandedMeasure. The word "this" makes reference to the CollapsedMeasure, which contains the de nition of the connection method. The value of expandedCompositeName attribute in the CollapsedMeasure should coincide with the name of the ExpandedMeasure pattern. If they do not match, responseMeasure is assigned as null.

```
if(this.getExpandedCompositeName().equals(pattern.getName()))
```

4. Check the validity. Both ExpandedMeasure and CollapsedMeasure to be connected should be valid. Validation of those measures were described at the beginning of this subsection. If at least one of them is not valid, responseMeasure is assigned as null.

```
if(pattern.valid() && this.valid())
```

5. Comparison (numbers). The number of connectors assigned to a CollapsedMeasure must be the same than the number of LeafMeasure identi ed for the definition of our ExpandedMeasure pattern measure. If they are not coincide responseMeasure is assigned as null.

```
if(this.getUsedConnectorMap().size()==
                        pattern.getLeafNodeMeasures().size())
```

6. Comparison (types). In addition to the number of elements, it is necessary to validate the type of the elements to be connected. An ExpandedMeasure has measures with particular connections depending on their types and a CollapsedMeasure has connectors with particular connections depending on their types. The number and types of connectors in a CollapsedMeasure must coincide with the number and types of measures in an ExpandedMeasure. Methods getNumberOfLeafMeasuresByTypes() and getNumberOfConnectorsByTypes() return

an object NumberOfTypes. Attributes of NumberOfTypes are: numAggregated, numCount, numData, numStateCondition, numTime, numList, numExternalValue and numOther to indicate the number of elements of each type that each CompositeMeasure has. If they do not coincide responseMeasure is assigned as null.

```
if(this.hasEqualTypeOfElements(
            pattern.getNumberOfLeafMeasuresByTypes(),
            this.getNumberOfConnectorsByTypes()) )
```

7. Clone. If previous conditions are ful lled, the pattern is clone to the new measure.

```
responseMeasure = pattern.clone();
```

8. Connecting measures. For each connector should exist a LeafMeasure where the connections will be copied. The search of the correct measure for each connector is made by means of the method connectMeasure that receives the connector to be copied and the complete structure of the measure de ned in the ExpandedMeasure. Because of each connector has different connections depending on its type, we have de ned one method for each type of connector (TimeConnector, CountConnector, DataConnector, etc.) but all of them has the same function: to nd a LeafMeasure that has the same type and name than the connector, to copy the value of each connection and return the complete structure of the measure. Then, this measure is assigned to the responseMeasure. After all connectors have been connected, the method return the complete ExpandedMeasure.

```
Object[] connectors = new Object[this.getUsedConnectorMap().size()];
connectors = this.getUsedConnectorMap().values().toArray();

for(int i = 0; i < connectors.length; i++){
  if(cleanString(connectors[i].getClass().getName()).
  contentEquals("TimeConnector")){
    responseMeasure.setMeasure(connectMeasure(
    (TimeConnector)connectors[i],responseMeasure.getMeasure()));
  }else{
    if(cleanString(connectors[i].getClass().getName()).
    contentEquals("CountConnector")){
      responseMeasure.setMeasure(connectMeasure(
      (CountConnector)connectors[i],responseMeasure.getMeasure()));
    }else
      //... Do the same for each type of connector
  }
}
```

9. Rename. Before to return the ExpandedMeasure connected, its name is changed by the name of the CollapsedMeasure used.

```
responseMeasure.setName(this.getName());
```

B.3 MODELLING A SCOR PATTERN USING THE EXTENSION OF PPINOT TOOL

Using the extension of the PPINOT core application, it is possible to define all patterns proposed in Table §4.2 and instantiate them by means of CompositeMeasure. In this section, we present an example to illustrate the utility of reuse. The pattern PercentageData is defined and reuse in two scenarios.

The PercentageData is calculated using the formula partial/total*100, where partial is the sum of the value property $< p\_dobjectproperty_1 >$ of the DataObject $< p\_dobjectname_1 >$ and total is the sum of the value property $< p\_dobjectproperty_2 >$ of the DataObject $< p\_dobjectname_2 >$. In our concrete example we calculate the percentage of errors in purchase orders due to incomplete orders.

First, an ExpandedMeasure should be defined with all measures required to calculate the reusable definition. The main measure is a DerivedMultiInstanceMeasure that uses partial and total measures. Those measures are two AggregatedMeasure calculated over two DataMeasure. The specific DataObject and the data object attribute will be defined in the CollapsedMeasure.

```
//1.- The expanded measure is defined

DataMeasure dmpartial = new DataMeasure();
dmpartial.setId("DM01");
dmpartial.setName("partial data");

DataMeasure dmtotal = new DataMeasure();
dmtotal.setId("DM02");
dmtotal.setName("total data");

AggregatedMeasure aggpartial = new AggregatedMeasure();
aggpartial.setId("AGM01");
aggpartial.setName("Sum of partial");
aggpartial.setAggregationFunction("SUM");
aggpartial.setBaseMeasure(dmpartial);

AggregatedMeasure aggtotal = new AggregatedMeasure();
aggtotal.setId("AGM01");
aggtotal.setName("Sum of total");
aggtotal.setAggregationFunction("SUM");
aggtotal.setBaseMeasure(dmtotal);

DerivedMultiInstanceMeasure dermpercentage =
                                new DerivedMultiInstanceMeasure();
dermpercentage.setId("DerM01");
dermpercentage.setName("DerM01-Percentage");
```

```
  dermpercentage.setFunction("aggpartial/aggtotal          *100");
  dermpercentage.addUsedMeasure(aggpartial.getName(), aggpartial);
  dermpercentage.addUsedMeasure(aggtotal.getName(), aggtotal);

  ExpandedMeasure exm = new ExpandedMeasure();
  exm.setId("EXP01");
  exm.setName("Percentage_Data");
  exm.setDescription("Calculates a percentage based on two data values");
  exm.setScale("-");
  exm.setUnitOfMeasure("number of orders");
  exm.setMeasure(dermpercentage);
```

To specify connections with the business process elements, a CollapsedMeasure is de ned. Here, two connectors are required to specify the partial and total part of the percentage. Depending on the PPI to be de ned, the data objects and properties should be speci ed. Each name connector should coincide with the name of the measure to be connected in the ExpandedMeasure and setExpandedCompositeName should coincide with the name of the ExpandedMeasure to be used.

```
  //2.- CollapsedMeasure is defined

  DataConnector ccpartial = new DataConnector();
  ccpartial.setId("CCData01");
  ccpartial.setName("partial data");
  ccpartial.setDescription("calculates the numerator of the function");
  ccpartial.setCcDataContentSelection(
                          new DataContentSelection("incomplete","purchase order"));

  DataConnector cctotal = new DataConnector();
  cctotal.setId("CCData02");
  cctotal.setName("total data");
  cctotal.setDescription("calculates the denominator of the function");
  cctotal.setCcDataContentSelection(
                          new DataContentSelection("created","purchase order"));

  CollapsedMeasure cllm = new CollapsedMeasure();
  cllm.setId("CLLO1");
  cllm.setName("Test of percentage data");
  cllm.setExpandedCompositeName("Percentage_Data");
  cllm.addUsedConnector(ccpartial.getName(), ccpartial);
  cllm.addUsedConnector(cctotal.getName(), cctotal);
```

Finally, the CollapsedMeasure (cllm) is connected with the ExpandedMeasure (exm) by means of the instruction

```
//3.- Connecting measures
ExpandedMeasure percentageIncompleteOrders =
                          cllm.connectCollapsedToExpanded(exm);
```

Additionally, if the connection is correct, the structure of the measure is printed.

```
if(percentageIncompleteOrders!=null)
    percentageIncompleteOrders.printAttributes();
```

The result of the method printAttributes() is shown bellow.

```
Expanded Measure :: Test of percentage data
- Id: EXP01
- Name: Test of percentage data
- Description: Measure that calculates a percentage based on two data values
- Scale: -
- Unit of measure: number of orders
- Measures and connections:
  DerivedMultiInstanceMeasure[DerM01-Percentage] connected with:
    AggregatedMeasure[Sum of total] aggregates:
      DataMeasure[total data]
      > DataContentSelection:: purchase order
    AggregatedMeasure[Sum of partial] aggregates:
      DataMeasure[partial data]
      > DataContentSelection:: purchase order
```

Now, if we want to de ne another measure, for example a measure that calculates the percentage of invoices paid, we reuse the de nition of the ExpandedMeasure (exm). We just need to de ne a new CollapsedMeasure and connect it with the exm as shown below.

```
 DataConnector ccpartial02 = new DataConnector();
 ccpartial02.setId("CCData0201");
 ccpartial02.setName("partial data");
 ccpartial02.setDescription("calculates the numerator of the function");
 ccpartial02.setCcDataContentSelection(
                          new DataContentSelection("paid","invoice"));

 DataConnector cctotal02 = new DataConnector();
 cctotal02.setId("CCData0202");
 cctotal02.setName("total data");
 cctotal02.setDescription("calculates the denominator of the function");
 cctotal02.setCcDataContentSelection(
                       new DataContentSelection("registered","invoice"));

 CollapsedMeasure cllm02 = new CollapsedMeasure();
 cllm02.setId("CLLO2");
 cllm02.setName("Percentage of invoices paid");
 cllm02.setExpandedCompositeName("Percentage_Data");
 cllm02.addUsedConnector(ccpartial02.getName(), ccpartial02);
 cllm02.addUsedConnector(cctotal02.getName(), cctotal02);

 ExpandedMeasure percentageErrorOrders =
                                    cllm02.connectCollapsedToExpanded(exm);
  if(percentageErrorOrders!=null)
    percentageErrorOrders.printAttributes();
```

and the result of the measure is:

```
Expanded Measure :: Percentage of invoices paid
- Id: EXP01
- Name: Percentage of invoices paid
- Description: Calculates a percentage based on two data values
- Scale: -
- Unit of measure: number of orders
- Measures and connections:
   DerivedMultiInstanceMeasure[DerM01-Percentage] connected with:
      AggregatedMeasure[Sum of total] aggregates:
         DataMeasure[total data]
         > DataContentSelection:: invoice
      AggregatedMeasure[Sum of partial] aggregates:
         DataMeasure[partial data]
         > DataContentSelection:: invoice
```

PPINOT Graphical Notation integrated to VBPML

This appendix has a double objective. The rst one is related to the formal de nition of PPINOT. Section §C.1 shows an example of PPI modelling using traditional formal definitions of PPINOT and its extended de nition considering variability, to compare the complexity and advantages of using one and the other. The second objective is related to the graphical notation of PPINOT. In Section §C.2 and Section §C.3 we describe how the graphical notation of PPINOT, Visual PPINOT can be extended to be integrated and used with other Variability Business Process Modelling Language (VBPML). As we explained in Section §5.5, we selected the most cited languages: one focused on the management by restriction, PROVOP; and the other one by extension, C-iEPC.

C.1 Defining PPIs using the PPINOT Formal Definitions

This section presents an example of a performance model based on the Deliver SCOR process and its measures. The example is divided into two parts: the rst one de- nes the performance model using the formal de nition of the original PPINOT metamodel; and the second one is constructed using the extended version of the PPINOT metamodel that includes the concepts of variability. The performance model is de ned according to the characteristics described below.

We focus on three of the process variants proposed in the SCOR model: Deliver Stocked Product (PV-1), Deliver Make-to-Order Product (PV-2) and Deliver Engineer-to-Order Product (PV-3). Those variants are associated with performing customer-facing order management and order ful llment activities.

SCOR de nes measures for each process variant. In this example, the performance model contains a single PPI based on the measure “Load Product and Generate Shipping Documentation Cycle Time”, which is related to the three process variants selected. It calculates the average time of product loading and the generation of shipping documentation. Table §C.1 shows the information required to de ne the PPI in each process variant.

Depending on the process variant where the PPI is de ned, it obtains information from one business process element or another. In this example, the time is measured from different activities of the process, which are speci ed in the rst row of Table §C.1.

Table C.1: Description of attributes to de ne a PPI in three PVs.

Name:	Load Product and Generate Shipping Documentation Cycle Time		
ID:	RS.3.51		
Attributes	**For PV-1**	**For PV-2**	**For PV-3**
Business Process Elements	Activity : Load Vehicle and Generate Shipping Documentation (LV&GSD)	Activity : Load Product and Generate Shipping Docs (LP&GSD)	Activity : Load Product and Generate Shipping Docs (LP&GSD)
Target:	Must be less than 4 hours (t_1)	Must be less than 3 hours (t_2)	Must be less than 3 hours (t_3)
Scope:	All instances (s_1)	All instances (s_2)	All instances (s_3)
Responsible:	Operator 1 (hr_{res1})	Operator 2 (hr_{res2})	Operator 2 (hr_{res3})
Informed:	Logistic Manager ($hr_{inf\,1}$)	Logistic Manager ($hr_{inf\,2}$)	Logistic Manager ($hr_{inf\,3a}$) General Manager of Production($hr_{inf\,3b}$)

The other rows (target, scope, responsible and informed) specify the PPI attributes required in a PPI de nition. From the values shown in Table §C.1 it is possible to identify variability in the PPI de nition, because some of the attributes vary depending on the process variant where the PPI is de ned. For example, since the activity where the measure is de ned changes, there is variability type Dim-2.1; and since the values of the target and the human resources assigned also change from one process variant to another, there is also variability type Dim-2.2. There is not variability of type Dim-1, because the PPI is related to the three process variants conforming the PF (PV-1, PV-2 and PV-3).

PPI de nitions using the formal de nition of the original PPINOT version.

Using this alternative, each process variant should be managed as an independent process, and the PPI (one or more) must be de ned separately for each variant. First, we specify the condition and then the performance model related to the PV is de ned.

De ning PPI RS.3.51 for PV-1:

- $C_{bp} = A \quad S_A \; [\; D \quad S_D \; [\; E \quad S_E$, where:
  $C_{PV \quad 1} = f\,(LV\&GSD, START), (LV\&GSD, END)\,g;$
  $bp = PV \quad 1; LV\&GSD\; 2\; A\; ; START, END\; 2\; S_A$
- $PM = (\,P, M, L_P, L_M\,)$, where:

  $PM_1 = (\,f\, RS.3.51g, f\, aggm_1, tm_1 g,$

f (RS.3.51,s_1), (RS.3.51,t_1), (RS.3.51,hr_{res1}), (RS.3.51,$hr_{inf\,1}$),
(RS.3.51,$aggm_1$) g,
f ($aggm_1$, tm_1, AVG), (tm_1, (LV&GSD, START)),
(tm_1, (LV&GSD, END)) g),

where:

- PM = PM_1;
- RS.3.512 P;
- f $aggm_1$, tm_1g M, where $aggm_1$ 2 AggM, tm_1 2 TimeM;
- f (RS.3.51,s_1), (RS.3.51,t_1), (RS.3.51,hr_{res1}), (RS.3.51,$hr_{inf\,1}$), (RS.3.51,$aggm_1$) g L_P, where:
  - (RS.3.51,s_1) 2 sco, where: s_1 2 S ;
  - (RS.3.51,t_1) 2 tar, where: t_1 2 T ;
  - (RS.3.51,hr_{res1}) 2 res, where: hr_{res1} 2 HR ;
  - (RS.3.51,$hr_{inf\,1}$) 2 inf , where: $hr_{inf\,1}$ 2 HR ;
  - (RS.3.51,$aggm_1$) 2 mes;
- f ($aggm_1$, tm_1, AVG), (tm_1, (LV&GSD, START)), (tm_1, (LV&GSD, END)) g L_M, where:
  - ($aggm_1$, tm_1, AVG) 2 agg;
  - (tm_1, (LV&GSD, START)) 2 from, where: LV&GSD 2 A and START 2 S_A ;
  - (tm_1, (LV&GSD, END)) 2 to;

De ning PPI RS.3.51 for PV-2:

- C_{bp} = A S $_A$ [D S $_D$ [E S $_E$, where:
  $C_{PV\ 2}$ = f (LP&GSD, START), (LP&GSD, END) g;
  bp = PV 2; LP&GSD 2 A ; START, END 2 S_A
- PM = (P, M, L_P, L_M), where:

  PM_2 =(f RS.3.51g, f $aggm_2$, tm_2g,
  f (RS.3.51,s_2), (RS.3.51,t_2), (RS.3.51,hr_{res2}), (RS.3.51,$hr_{inf\,2}$),
  (RS.3.51,$aggm_2$) g,
  f ($aggm_2$, tm_2, AVG), (tm_2, (LP&GSD)), (tm_2, (LP&GSD, END)) g),

  where:

  - PM = PM_2;
  - RS.3.512 P;

- f $aggm_2, tm_2$g   M, where $aggm_2$ 2 AggM, tm_2 2 TimeM;
- f (RS.3.51, s_1), (RS.3.51, t_2), (RS.3.51, hr_{res2}), (RS.3.51, $hr_{inf\,2}$), (RS.3.51, $aggm_2$) g L_P, where:
  - ⋆ (RS.3.51, s_2) 2 sco, where: s_2 2 S ;
  - ⋆ (RS.3.51, t_2) 2 tar, where: t_2 2 T ;
  - ⋆ (RS.3.51, hr_{res2}) 2 res, where: hr_{res2} 2 HR ;
  - ⋆ (RS.3.51, $hr_{inf\,2}$) 2 inf , where: $hr_{inf\,2}$ 2 HR ;
  - ⋆ (RS.3.51, $aggm_2$) 2 mes;
- f ($aggm_2$, tm_2, AVG), (tm_2, (LP&GSD, START)), (tm_2, (LP&GSD, END)) g L_M, where:
  - ⋆ ($aggm_2$, tm_2, AVG) 2 agg;
  - ⋆ (tm_2, (LP&GSD, START)) 2 from, where: LP&GSD 2 A and START 2 S_A ;
  - ⋆ (tm_2, (LP&GSD, END)) 2 to;

De ning PPI RS.3.51 for PV-3:

- C_{bp} = A S $_A$ [D S $_D$ [E S $_E$, where:

  $C_{PV\ \ 3}$ = f (LP&GSD, START), (LP&GSD, END) g;

  bp = PV 3; LP&GSD 2 A ; START, END 2 S_A

- PM = (P, M, L_P, L_M), where:

  PM_3 =(f RS.3.51g, f $aggm_3$, tm_3g,
  f (RS.3.51, s_3), (RS.3.51, t_3), (RS.3.51, hr_{res3}), (RS.3.51, $hr_{inf\,3a}$),
  (RS.3.51, $hr_{inf\,3b}$), (RS.3.51, $aggm_3$) g,
  f ($aggm_3$, tm_3, AVG), (tm_3, (LP&GSD)), (tm_3, (LP&GSD, END)) g),

  where:

  - PM = PM_3;
  - RS.3.512 P;
  - f $aggm_3$, tm_3g   M, where $aggm_3$ 2 AggM, tm_3 2 TimeM;
  - f (RS.3.51, s_3), (RS.3.51, t_3), (RS.3.51, hr_{res3}), (RS.3.51, $hr_{inf\,3a}$), (RS.3.51, $hr_{inf\,3b}$), (RS.3.51, $aggm_3$) g L_P, where:
    - ⋆ (RS.3.51, s_3) 2 sco, where: s_3 2 S ;
    - ⋆ (RS.3.51, t_3) 2 tar, where: t_3 2 T ;
    - ⋆ (RS.3.51, hr_{res3}) 2 res, where: hr_{res3} 2 HR ;
    - ⋆ f (RS.3.51, $hr_{inf\,3a}$), (RS.3.51, $hr_{inf\,3b}$) g   inf , where: f $hr_{inf\,3a}$, $hr_{inf\,3b}$g 2 HR ;

    - $\ast$ $(RS.3.51, aggm_3)$ 2 mes;
  - f $(aggm_3, tm_3, AVG), (tm_3, (LP\&GSD, START)), (tm_3, (LP\&GSD, END))$ g L_M, where:
    - $\ast$ $(aggm_3, tm_3, AVG)$ 2 agg;
    - $\ast$ $(tm_3, (LP\&GSD, START))$ 2 $from$, where: $LP\&GSD$ 2 A and $START$ 2 S_A;
    - $\ast$ $(tm_3, (LP\&GSD, END))$ 2 to;

PPI de nitions using the formal de nition of the extended PPINOT version.

The following example represents the same scenario described before: a PPI RS.3.51 that is de ned differently for three variants (PV-1, PV-2, PV-3) of the Deliver SCOR process (see Table §C.1); but this time using the extended de nition of PPINOT. Using this alternative, a process family and a performance model of variability must be de ned.

- PF = f $bp_1, ..., bp_n$ g = f PV-1, PV-2, PV-3g is the set of Deliver process variants.
- PF = $P(PF)$ n ? = f f PV-1g, f PV-2g, f PV-3g, f PV-1, PV-2g, f PV-1, PV-3g, f PV-2, PV-3g, f PV-1, PV-2, PV-3g g;
- $PM^V = (P, M, L_P, L_M, P^V, L_P^V, L_M^V)$ =
  - P, M, L_P, L_M refer to elements of a performance model de ned over C_{PF};
    - $\ast$ $RS.3.51$ 2 P
    - $\ast$ f $aggm, tm$ g 2 M, where $aggm$ 2 $AggM$ and tm 2 $TimeM$;
    - $\ast$ f $(RS.3.51, s_1), (RS.3.51, s_{23}), (RS.3.51, t_1), (RS.3.51, t_{23}), (RS.3.51, hr_{res1}), (RS.3.51, hr_{res23}), (RS.3.51, hr_{inf\,123}), (RS.3.51, hr_{inf\,3b})$ g 2 L_P, where:
      - · f $(RS.3.51, s), (RS.3.51, s)$ g sco, where: f s g S;
      - · f $(RS.3.51, t_1, (RS.3.51, t_{23}$ g tar, where: f t_1, t_{23} g T;
      - · f $(RS.3.51, hr_{res1}), (RS.3.51, hr_{res23})$ g res, where: f hr_{res1}, hr_{res23} g HR;
      - · f $(RS.3.51, hr_{(inf\,123a)}, (RS.3.51, hr_{inf\,3b})$ g inf, where: f $hr_{inf\,123a}, hr_{inf\,3b}$ g HR;
    - $\ast$ f $(RS.3.51, aggm), (tm, (LV\&GSD, START)), (tm, (LP\&GSD, START)), (tm, (LV\&GSD, END)), (tm, (LP\&GSD, END)), (aggm, tm, AVG)$ g 2 L_M, where:
      - · $(RS.3.51, aggm)$ 2 mes;
      - · f $(tm, (LV\&GSD, START)), (tm, (LP\&GSD, START))$ g $from$;
      - · f $(tm, (LV\&GSD, END)), (tm, (LP\&GSD, END))$ g to;
      - · $(aggm, tm, AVG)$ 2 agg, where: AVG 2 F_{agg};

- $\{(LV\&GSD, START), (LP\&GSD, START), (LV\&GSD, END), (LP\&GSD, END)\} \quad C$;
- $\{LV\&GSD, LP\&GSD\} \quad A$;
- $\{START, END\} \in S_A$;

– $P^V : P \to PF \quad = \{RS.3.51\} \mapsto \{PV\text{-}1, PV\text{-}2, PV\text{-}3\}$;

$$-L_P^V = L_P \to PF \quad =$$
$$\{\{(RS.3.51, s), (RS.3.51, t_1), (RS.3.51, hr_{a1})\} \mapsto \{PV\text{-}1\} \cup$$
$$\{(RS.3.51, s), (RS.3.51, t_{23}), (RS.3.51, hr_{a23})\} \mapsto \{PV\text{-}2, PV\text{-}3\} \cup$$
$$\{(RS.3.51, hr_{(1b)}), (RS.3.51, aggm)\} \mapsto \{PV\text{-}1, PV\text{-}2, PV\text{-}3\} \cup$$
$$\{(RS.3.51, hr_{(2b)3})\} \mapsto \{PV\text{-}3\}\};$$

$$-L_M^V = L_M \to PF \quad =$$
$$\{\{(tm, (LV\&GSD, START)), (tm, (LV\&GSD, END))\} \mapsto \{PV\text{-}1\} \cup$$
$$\{(tm, (LP\&GSD, START)), (tm, (LP\&GSD, END))\} \mapsto \{PV\text{-}2, PV\text{-}3\} \cup$$
$$\{(aggm, tm, AVG)\} \mapsto \{PV\text{-}1, PV\text{-}2, PV\text{-}3\}\};$$

In the definition of L_P^V we have changed the naming of PPI attributes and we have named them depending on the process variants where they are used, as follows: the scope value is the same for the three process variants, originally named as s_1, s_2, s_3 and they were replaced by s; for the target, t_2 and t_3 have the same values and were replaced by s_{23} to indicate that the two process variants share the value. The same occurs with human resources. hr_a represents human resource Responsible, which for PV-1 is still hr_{res1}, but hr_{res2} and hr_{res3} are replaced by hr_{res23}.

Comments about the examples

Both alternatives allow the definition of performance models for different PVs. In the first alternative each PV is managed individually, thus generating a large amount of redundant data, since attributes such as Scope or the human resource Informed have the same value for all PVs. In addition, if a change is required in one or several attributes, each change must be implemented individually, which can lead to errors. For example, if instead of “all instances” the scope value varies to “the last 30 instances”, the three variants must be modified individually. In the second alternative, managing all PVs as a collection reduces redundant information, because each value is associated with a set of PVs. In the same way, changes of values associated with the performance model requires less effort because changes are centralised and applied directly to all PVs related to that value.

Figure C.1: Change patterns used in the PROVOP approach to modify a base process model

C.2 Using Visual PPINOT with PROVOP

In this section, we describe how Visual PPINOT can be integrated with a VBPML. We have selected PROVOP as a VBPML that manages variability by restriction, but in fact, this approach, as other ones able to manage variability by extension, allows the management of variability by extension and by restriction.

The PROVOP approach manages variability on the basis of a process model that gather certain characteristics of the process family. Several criteria can be selected to de ne this base model. In most cases, the base model is built with the common parts of all or most of the process variants.

To con gure a process variant from the base model, a set of options is proposed. Each option contains a sequence of instructions that indicates part of the business process that will be modi ed. These instructions are represented by change patterns: INSERT fragment, DELETE fragment, MOVE fragment, and MODIFY attribute. The rst three patterns may be applied to a model fragment and the latter pattern can be used to modify the value of a process element attribute. Those patterns are depicted as shown in Figure §C.1.

In line with the original approach, we have de ned a set of change patterns, to indicate how PPIs de ned on a PROVOP base model should be modi ed. Similarly, we start from a base PPI, which can represent an indicator with the most repeated characteristics in all the variants of the process, for example.

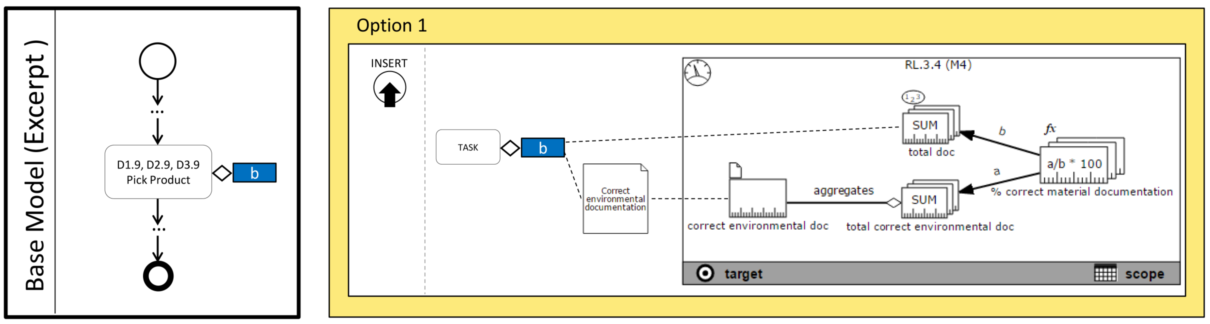


Figure C.2: Example of Change patterns to Insert a PPI in a PROVOP base model

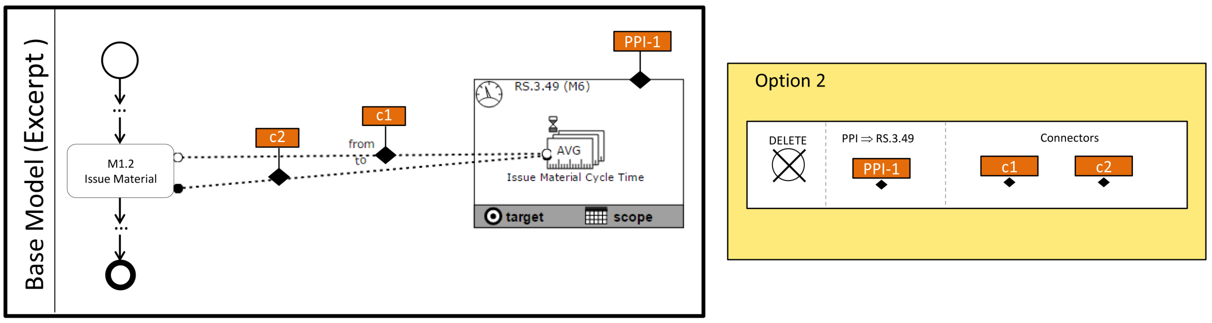


Figure C.3: Example of Change patterns to Delete a PPI in a PROVOP base model

The sequence of instruction to modify the base model are also specified by means of the following change patterns:

- **INSERT PPI**: Given a set of adjustment points, it indicates where in the base model a given PPI should be connected. The PPI to inserted is defined as an instruction of an option, and the PPI is connected using a set of adjustment points. If specific characteristics need to be applied over the PPI, e.g. new values for its attributes, the MODIFY patter should be used. This is explain in further items. Figure §C.2 shows an example of the INSERT change pattern.

- **DELETE PPI**: Given a PPI defined in a base model, we can indicate the PPI that should be deleted. It is required to indicate by means of adjustment points the PPI and connectors to be deleted. In Figure §C.3 we show an example using this change pattern.

- **MOVE PPI**: This change patterns is used connect the PPI to a different set of business process elements of the process model. A set of adjustment points need to be specified to indicate the current points of connection and the new points where the PPI is going to be connected. Figure §C.4 shows an example using the change pattern to move a PPI from one point to another.

- **MODIFY PPI Attribute**: As in the original version of PROVOP, the change pattern MODIFY affects an attribute, in this case, an attribute of the PPI. It is necessary to indicate the PPI to which it affects, the attribute to be modified and the

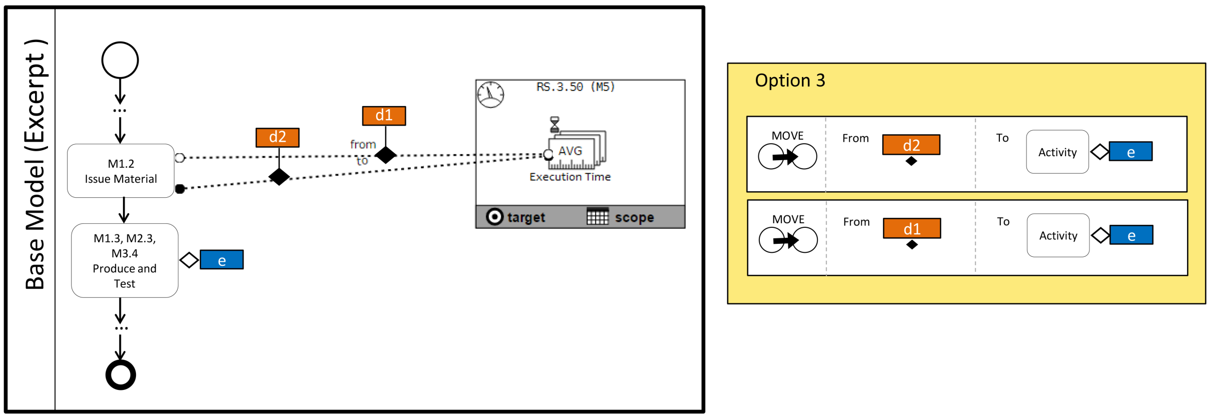


Figure C.4: Example of Change patterns to Move a PPI from one place to another one in the PROVOP base model

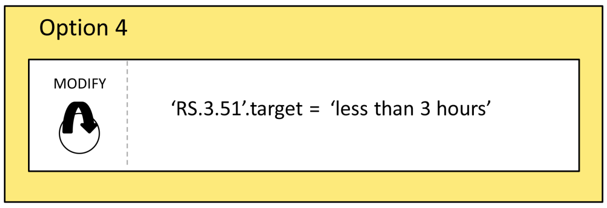


Figure C.5: Example of Change patterns to Modify a PPI from one place to another one in the PROVOP base model

new value of the attribute to be assigned. Figure §C.5 shows an example using the change pattern MODIFY to change the attribute value of a PPI. In this example, the PPI to be modified is the PPI 'RS.3.51' and the attribute to be modified is the 'target'.

These change patterns can be used in combination with the change patterns describe to modify the base model (shown in Figure §C.1) or they can be grouped into a stand-alone option that works in combination with other options, as used in the example in Chapter §5.5.2.

C.3 Using Visual PPINOT with C-iEPC

In this section, we describe how Visual PPINOT can be integrated with the C-iEPC VBPML that manages variability by restriction. For this approach, all process variants are gathered in a single process model called *configurable integrated process model*, hereinafter *configurable process* for short.

A process variant is C-iEPC is derived from a configurable process by means of a set of restrictions and guidelines applied to configurable process elements of the process. Configurable functions are applied over configurable nodes of the process. Although

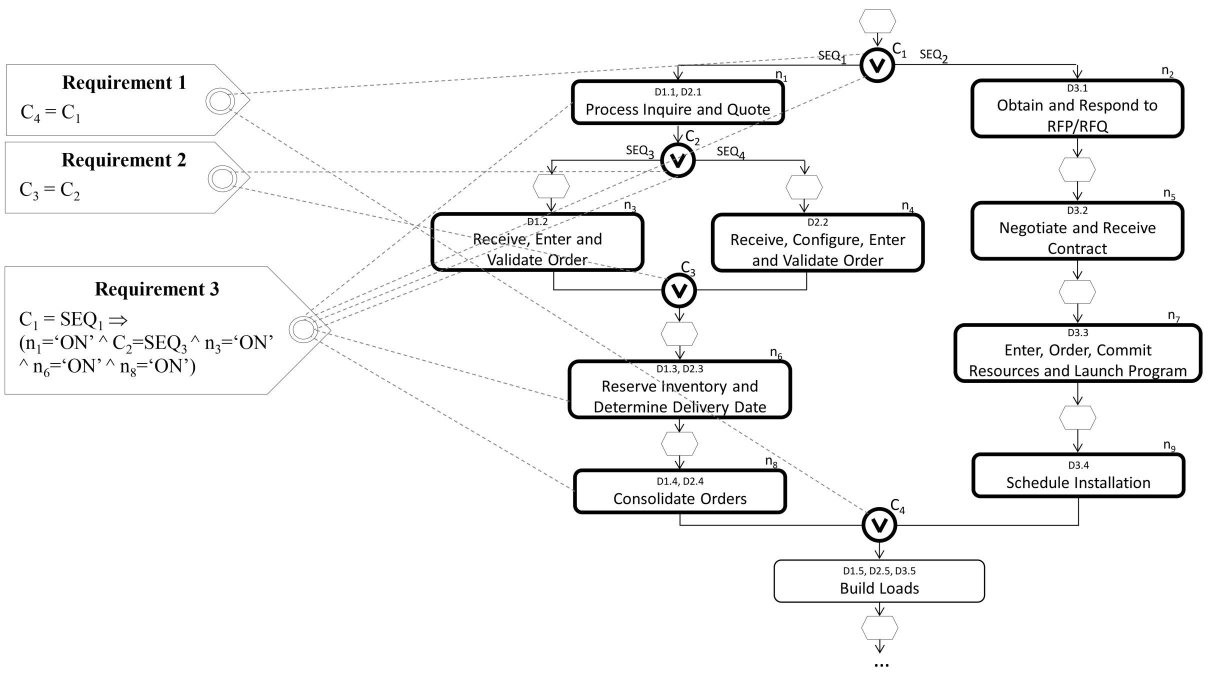

Requirement 1
C4 = C1
Requirement 2
C3 = C2
Requirement 3
C1 = SEQ1 ⇒
(n1='ON' ^ C2=SEQ3 ^ n3='ON'
^ n6='ON' ^ n8='ON')
SEQ1
C1
SEQ2
n1
D1.1, D2.1
Process Inquire and Quote
n2
D3.1
Obtain and Respond to RFP/RFQ
C2
SEQ3
SEQ4
n3
D1.2
Receive, Enter and Validate Order
n4
D2.2
Receive, Configure, Enter and Validate Order
n5
D3.2
Negotiate and Receive Contract
C3
n7
D3.3
Enter, Order, Commit Resources and Launch Program
n6
D1.3, D2.3
Reserve Inventory and Determine Delivery Date
n8
D1.4, D2.4
Consolidate Orders
n9
D3.4
Schedule Installation
C4
D1.5, D2.5, D3.5
Build Loads
...

Figure C.6: Excerpt of the C-iEPC configurable model to illustrate how a process variant can be derived

a complete C-iEPC model allows us to consider resources and objects assigned to activities; in this examples we focused on configurable nodes and activities.

In Figure §C.6, we show an excerpt of a configurable process to illustrate how a process variant can be derived. Activities with a thicker border line represent configurable nodes that may or may not be part of a process variant, depending on the constraint defined. In this case, requirement 3 indicates how to define a process variant.

As with process activities, it is also necessary to define all possible PPIs that will be involved in all variants of the process. In some cases the PPI is defined for some or all process variants. If the PPI is connected to an activity that is involved in the process variant, the PPI is linked to that variant. If the activity to which the PPI is connected does not participate in the process variant, the indicator does not participate either.

In order to be able to define which activities, configurable or not, a PPI is related to, a notation has been defined that allows this relationship to be established and highlighted. Figure §C.7 shows a PPI (RS.3.51) that is related to three process variant. The first process variant contains the Activity 'Load Vehicle and Generate Shipping Documents', while the other two process variants involve the Activity 'Load Product & Generate Shipping Docs'. To graphically represent the process variants to which the PPIs are related, Visual PPINOT is extended and variability connectors are used. The extended notation is described below.

We distinguish four categories of elements to depict variability in Visual PPINOT:

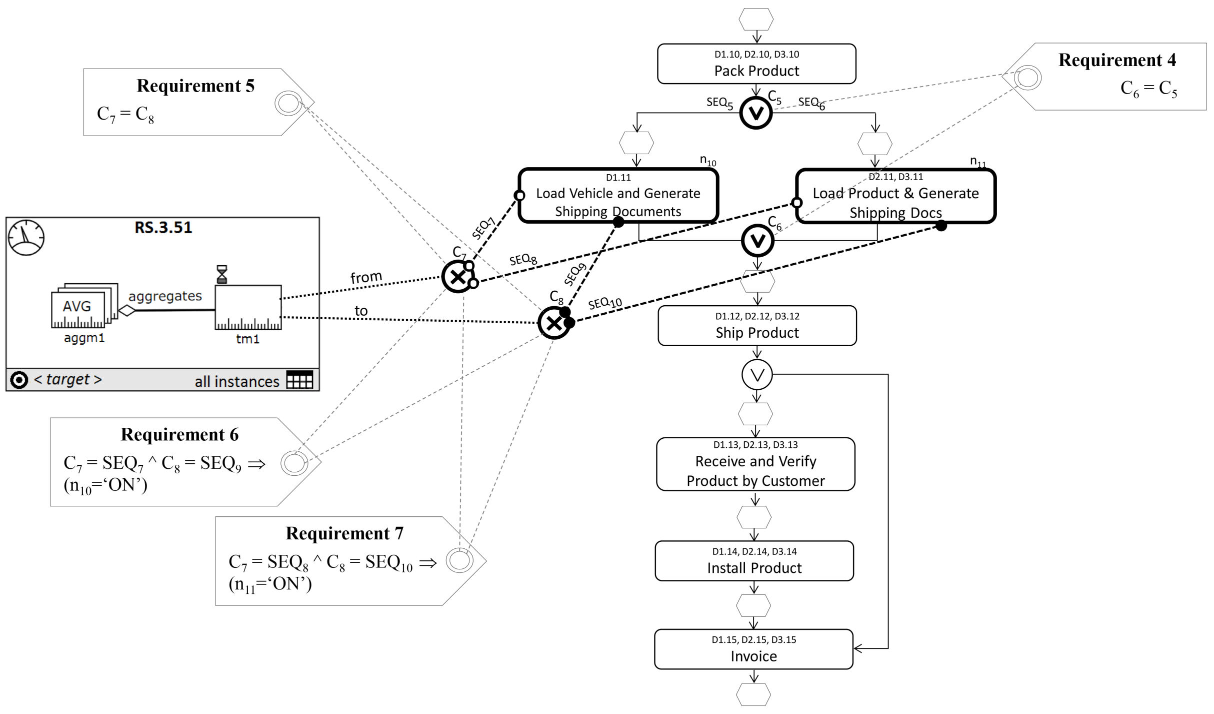


Figure C.7: Excerpt of the C-iEPC configurable model with a PPI defined for more than one process variant

- ***Variant connectors*** are connectors that allow us to specify behaviours, similar to gateways in BPMN. This elements are used to connect *PPIVariantsPoints* to `PPIVariants`, `MeasureVariationPoints` to `MeasureVariants`, or `PPIs` and measures without variability, as part of a variable element. There are four types of variant connectors depending on the behaviour of the flow.

  - *Exclusive* routes the sequence flow to one of the variant flows.
  - *Parallel* indicates that all outgoing variant flows are used.
  - *Parallel inclusive* routes the sequence flow to one or more variant flows.
  - *Complex* is used when behaviour is not captured by other connectors.

- ***Measures.*** All the original Visual PPINOT measures can be used in a scenario with variability. In addition there are two new types of variability measures. `MeasureVariationPoint` is defined as a new type of `MeasureDefinition`. It is composed by more than one `MeasureVariant`, representing all different ways to define a measure. Graphical representation of `MeasureVariant` is similar to graphical representation of PPINOT measures, but new ones have a hexagon wrapping the type of symbol measure at the top of it. A `MeasureVariant` can be related to different measures or objects in accordance with the original restrictions of PPINOT.

- ***PPIs*** are divided in two types:

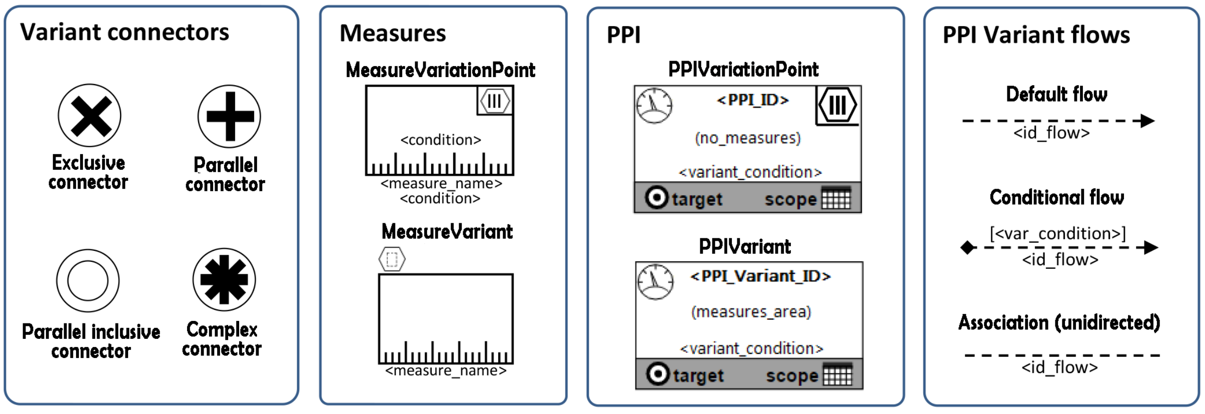


Figure C.8: Extension of the Visual PPINOT Notation taking into account variability.

- `PPIVariationPoint` indicates a PPI can be defined in more than one way (with different attributes like scope, target, human resources or measures). Graphically it is differentiated by a hexagon in the upper-right corner with 3 vertical bars inside. A `PPIVariationPoint` can only be related to `PPIVariants`, using PPI connectors for it; furthermore, this type of PPI can not contain measures inside.
- A `PPIVariant` represents one of the different ways in which a PPI can be calculated. It is similar to the original PPI without variability used in original version of Visual PPINOT.

• ***PPI Variant flows*** are used to link a `PPIVariationPoint` or `MeasureVariationPoint` with their corresponding `PPIsVariants` or `MeasureVariant`. Each flow has an implicit or explicit restriction to indicate process variants in which they may be applied. There are 3 types:

- *Default-flow* is a pointing arrow. Its restriction is implicit, assumed as the existence of the object bound.
- *Conditional flow* is a pointing arrow with a diamond in its starting point. Its restriction is explicit and is shown at the top of the arrow.
- *Association (unidirected)* is use to link a `PPIVariationPoint` to *Variant-connectors*, a *Variant-connector* to `PPIVariants`, a *Variant-connector* to measures or *Variant-measures*.

The following Figures §C.9 and §C.10, we graphically show the relationship between the new elements of notation. Figures §C.9 shows the relationship between a `PPIVariationPoint` that represents a PPI that can be implemented in more than one way for more than one PPI variant. As we are using an *exclusive variant connector* it indicates that a condition is needed to indicate what variant of the PPI is going to be applied in a specific process variant. Figure §C.10 shows all possible relationships between the new elements of the notation and the original PPINOT elements.

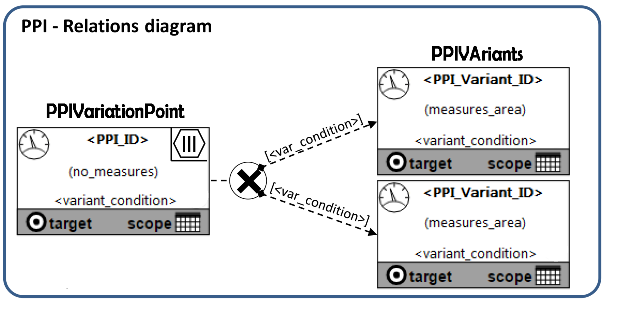


Figure C.9: Relationships between PPIs in the extension of the Visual PPINOT Notation

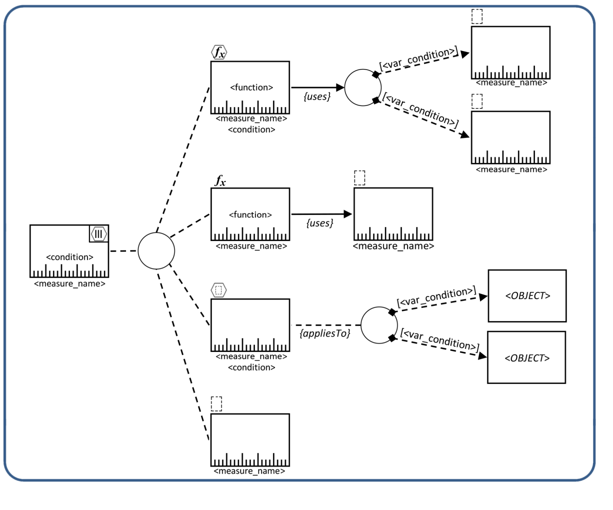


Figure C.10: Relationships between new elements of the Visual PPINOT Notation considering variability.

PPIs modelled using KiPPINOT

In this section we present the PPIs identified in the case study presented in Section §6.5.

D.1 Modelling of existing PPIs implemented by the ICT company using KiPPINOT.

In this section we present the set of PPIs considered in the first step of the methodology applied in the case study presented in Section §6.5. The set of PPIs considered in this section are:

- Average work time spent per ticket (Work Time AVG) - Figure §D.1.
- Average duration of tickets (Duration AVG) - Figure §D.2.
- First contact resolution ratio (First Cont. Res. %) - Figure §D.3.
- Total tickets opened (Total Tickets) - Figure §D.4.

D.2 Modelling Lead Indicators of a case study using KiPPINOT.

In this section we present the set of PPIs considered in the second step of the methodology applied in the case study presented in Section §6.5. The set of PPIs considered in this section are:

- Average amount of interlocutors per case (Interlocutor AVG) - Figure §D.5.
- Average amount of messages exchanged per case (Messages Exch AVG) - Figure §D.6.
- Average message size per case (Message Size AVG) - Figure §D.7.
- Incoming customer phone call ratio (Phone Call Ratio) - Figure §D.8.

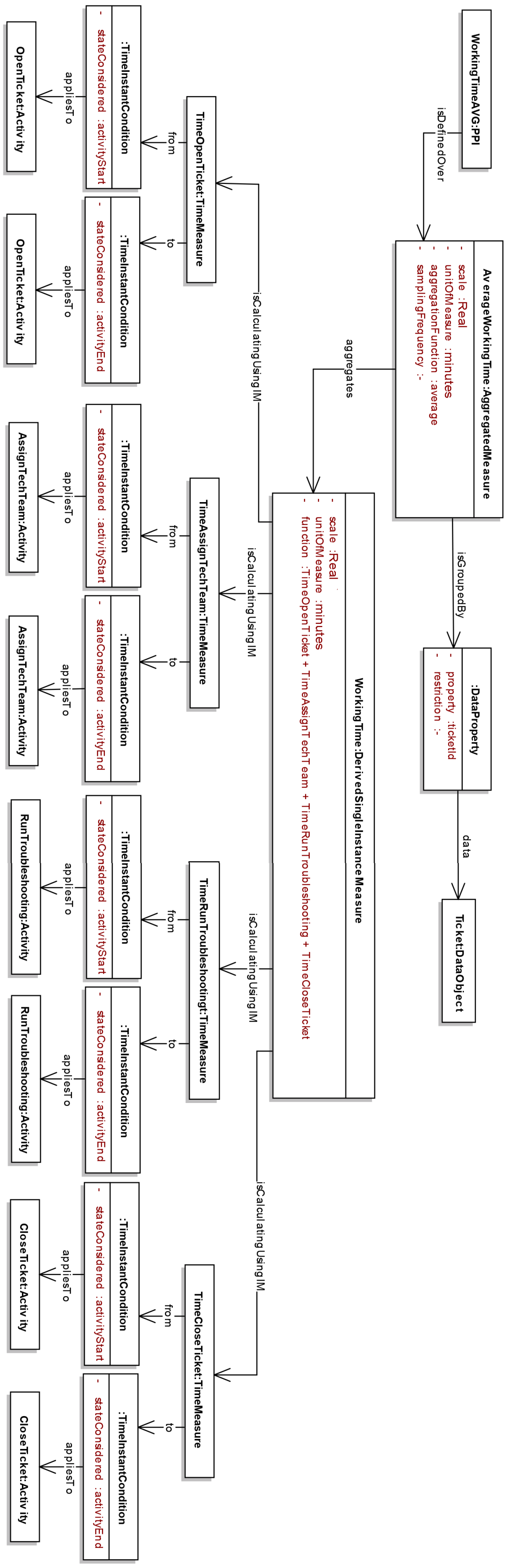


Figure D.1: Average work time spent per ticket (Work Time AVG). The PPI calculates the time spent by technical teams for solving a ticket incident.

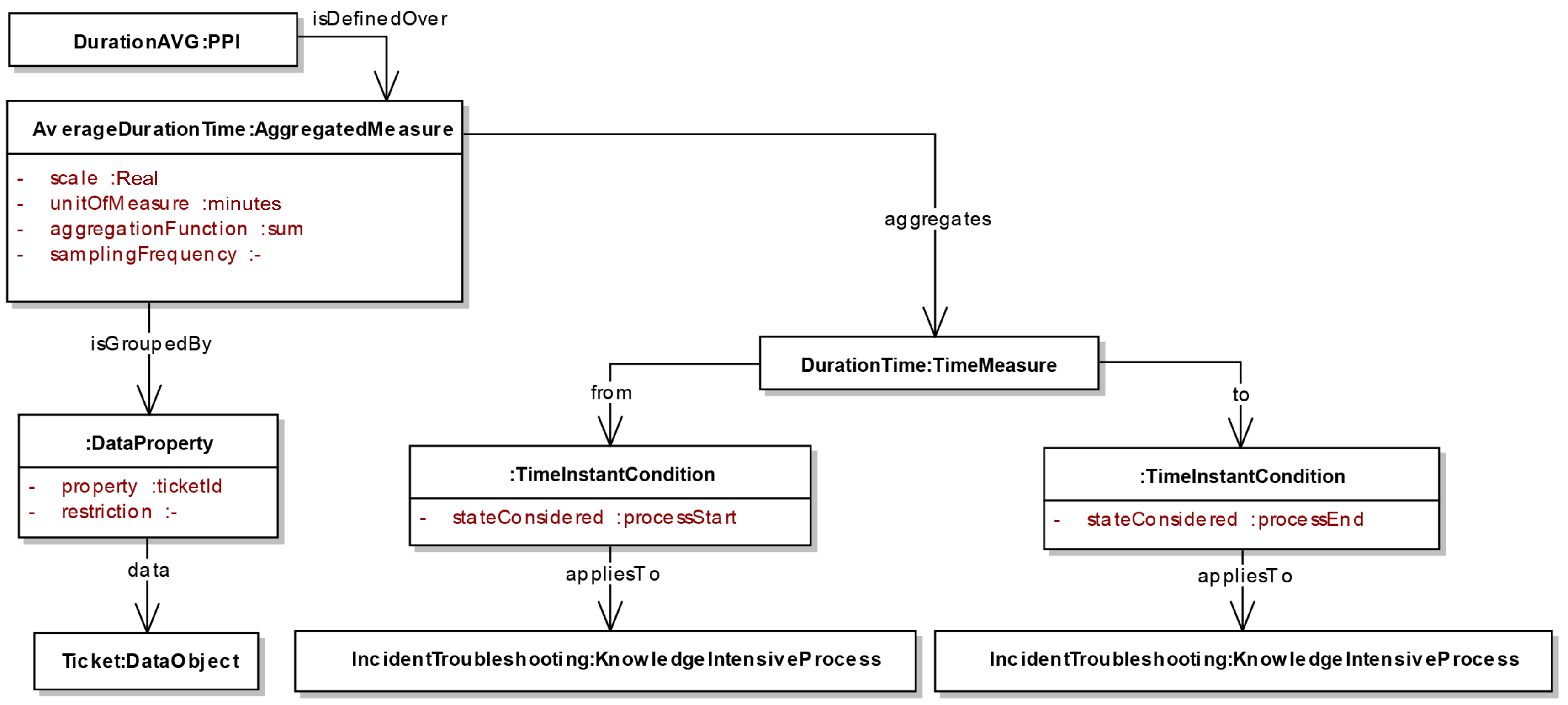


Figure D.2: Average duration of tickets (Duration AVG). This PPI measures the time from a ticket opening to its closure. The whole process is measured because waiting times and delays are considered.

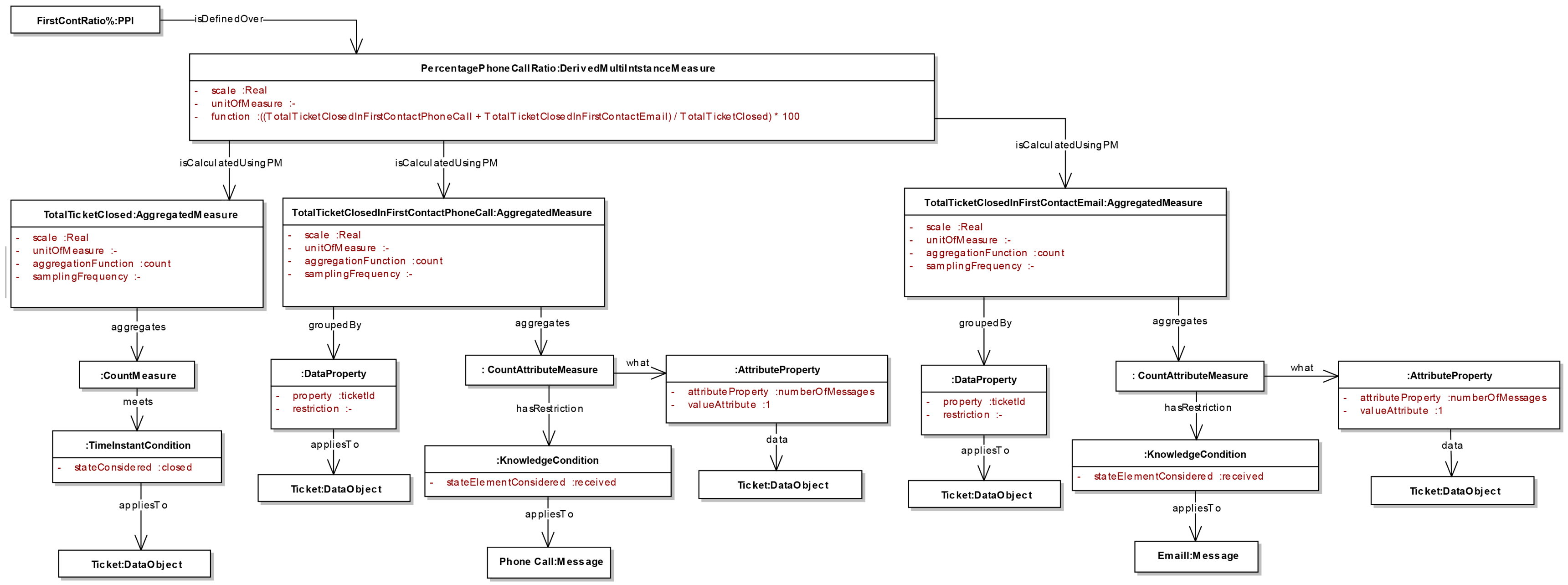


Figure D.3: First contact resolution ratio (First Cont. Res. %). This PPI measures the ratio of tickets that were solved directly in the first contact support teams.%)

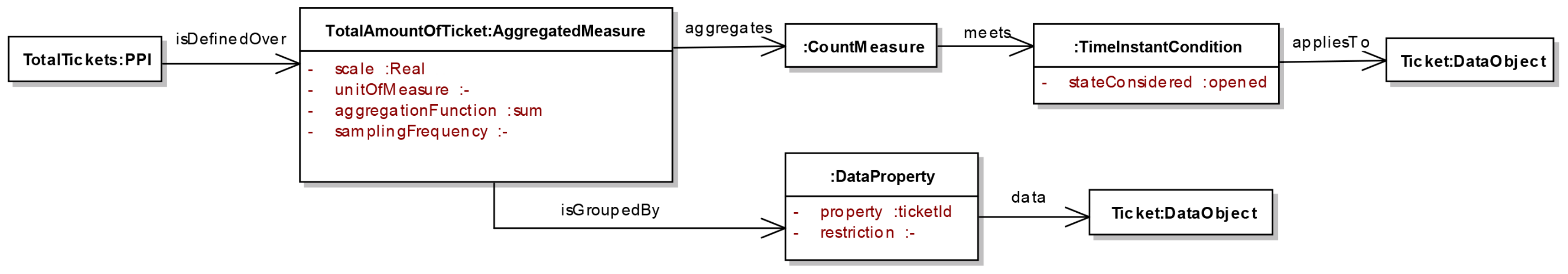


Figure D.4: Total tickets opened (Total Tickets). PPI that calculates the amount of tickets opened for each client. An aggregated measure is used to consider all process instances.

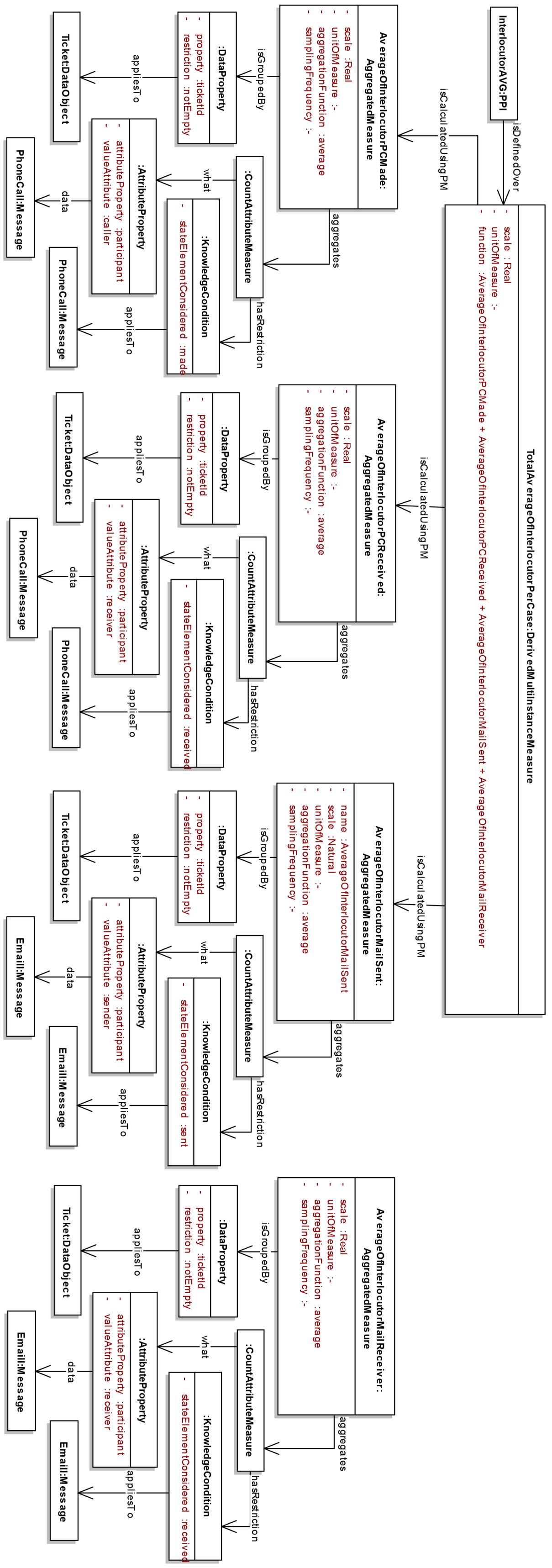

InterlocutorAVG:PPI
isDefinedOver
TotalAverageOfInterlocutorPerCase:DerivedMultiInstanceMeasure
- scale : Real
- unitOfMeasure :-
- function : AverageOfInterlocutorPCMade + AverageOfInterlocutorPCReceived + AverageOfInterlocutorMailSent + AverageOfInterlocutorMailReceiver
isCalculatedUsingPM
AverageOfInterlocutorPCMade: AggregatedMeasure
- scale : Real
- unitOfMeasure :-
- aggregationFunction : average
- samplingFrequency :-
AverageOfInterlocutorPCReceived: AggregatedMeasure
AverageOfInterlocutorMailSent: AggregatedMeasure
- name : AverageOfInterlocutorMailSent
- scale : Natural
AverageOfInterlocutorMailReceiver: AggregatedMeasure
isGroupedBy
aggregates
:DataProperty
- property : ticketId
- restriction : notEmpty
:CountAttributeMeasure
what
hasRestriction
:KnowledgeCondition
- stateElementConsidered : made
- stateElementConsidered : received
- stateElementConsidered : sent
:AttributeProperty
- attributeProperty : participant
- valueAttribute : caller
- valueAttribute : receiver
- valueAttribute : sender
appliesTo
data
Ticket:DataObject
PhoneCall:Message
Email:Message

Figure D.5: Average amount of interlocutors per case related to tickets resolution (Interlocutor AVG). This PPI measures how many agents participate in communicative acts. Those participants are directly related to Tickets.

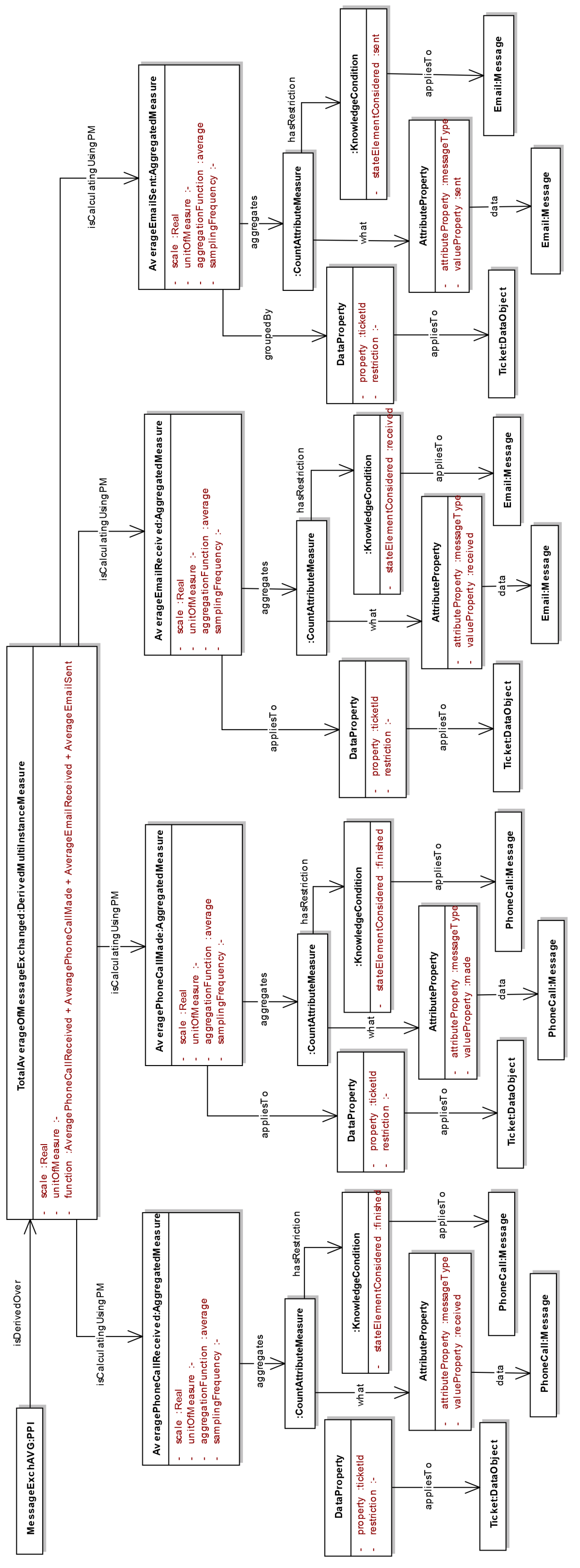

MessageExchAVG:PPI
isDerivedOver
TotalAverageOfMessageExchanged:DerivedMultiInstanceMeasure
scale :Real
unitOfMeasure :-
function :AveragePhoneCallReceived + AveragePhoneCallMade + AverageEmailReceived + AverageEmailSent
isCalculatingUsingPM
AveragePhoneCallReceived:AggregatedMeasure
AveragePhoneCallMade:AggregatedMeasure
AverageEmailReceived:AggregatedMeasure
AverageEmailSent:AggregatedMeasure
scale :Real
unitOfMeasure :-
aggregationFunction :average
samplingFrequency :-
aggregates
appliesTo
groupedBy
:CountAttributeMeasure
hasRestriction
what
:KnowledgeCondition
stateElementConsidered :finished
stateElementConsidered :received
stateElementConsidered :sent
AttributeProperty
attributeProperty :messageType
valueProperty :received
valueProperty :made
valueProperty :sent
DataProperty
property :ticketId
restriction :-
data
Ticket:DataObject
PhoneCall:Message
Email:Message

Figure D.6: Average amount of messages exchanged per case (Message Exch AVG). The PPI measures the number of messages exchanged during each ticket resolution.

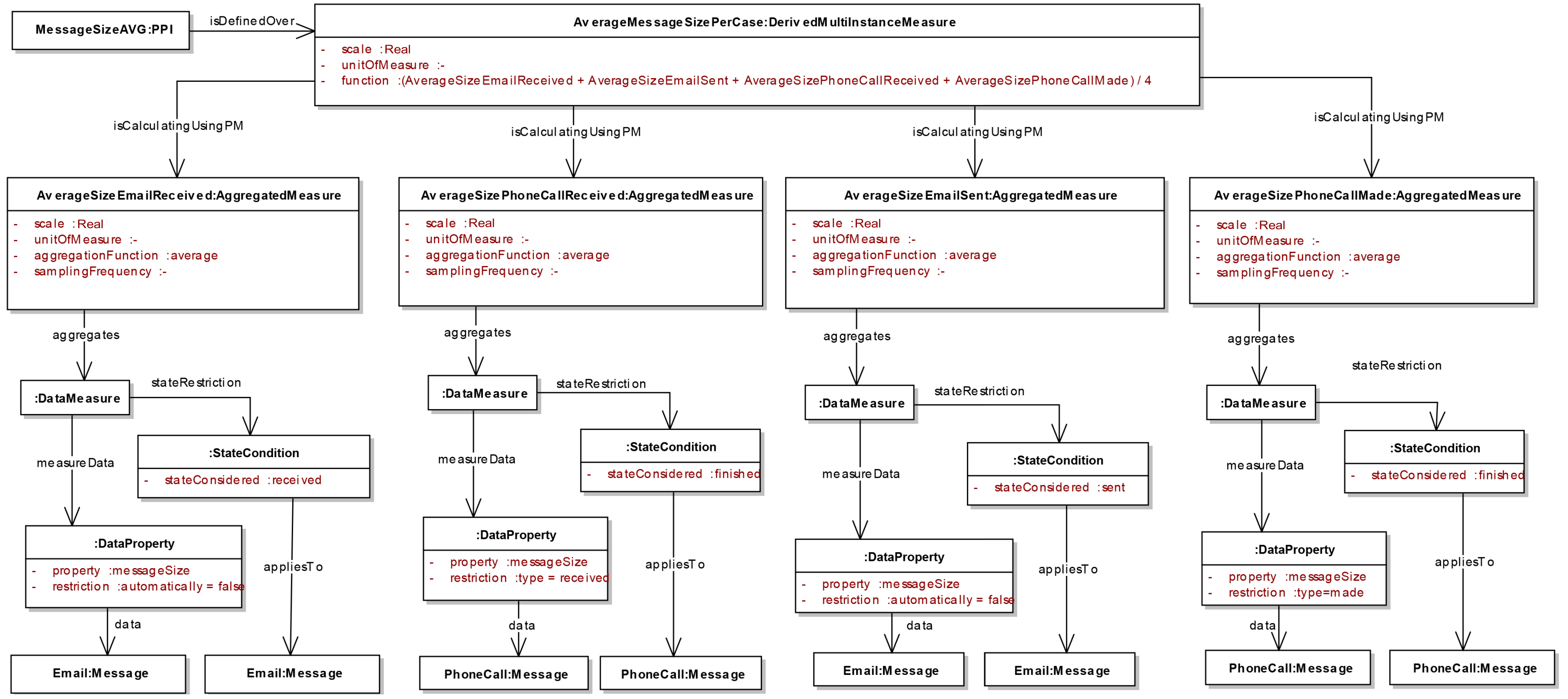


Figure D.7: Average message size per case (Message Size AVG). The PPI calculates the average size of messages exchanged between customer and members of the technical team. Messages could be emails or phone calls.

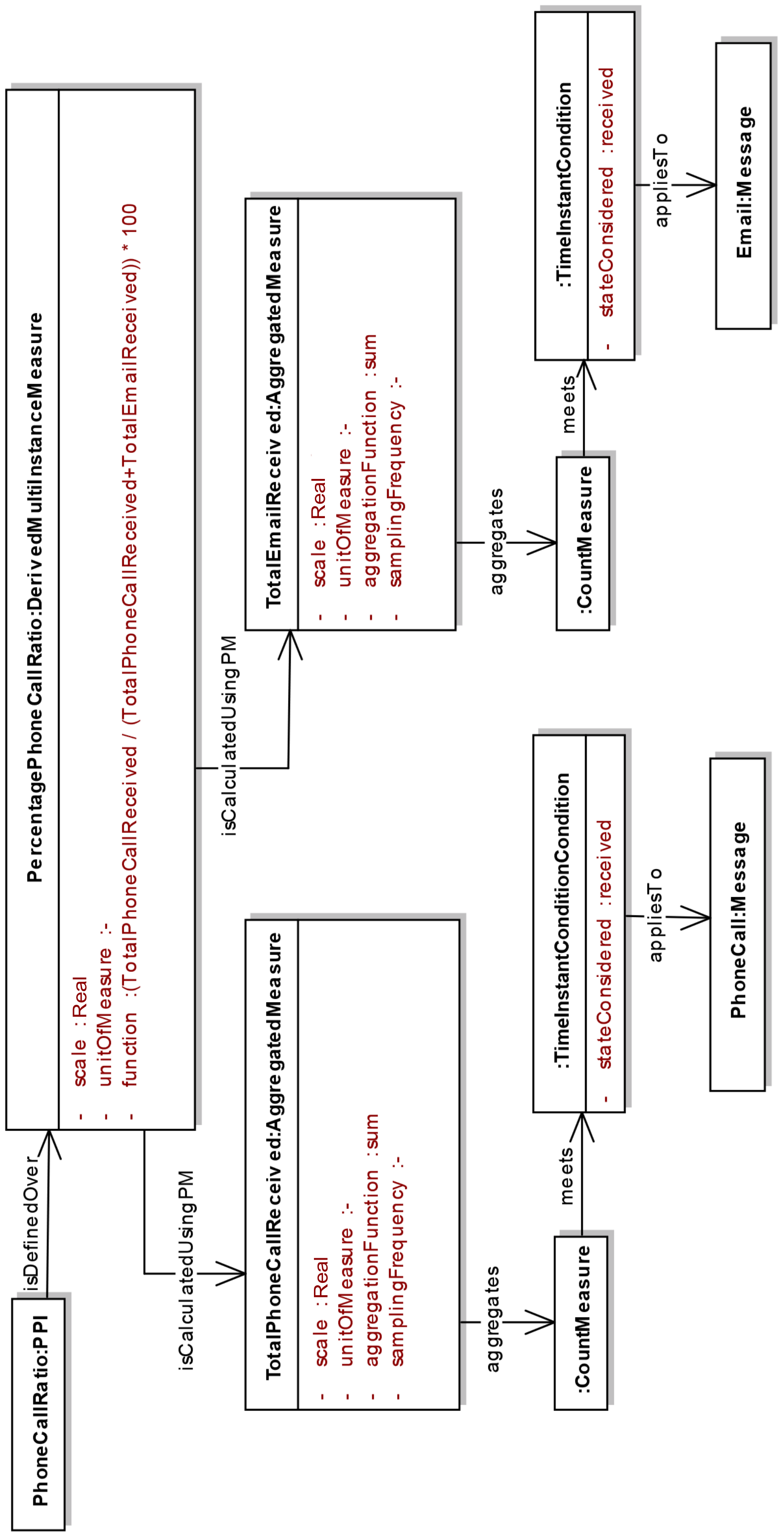

PhoneCallRatio:PPI
isDefinedOver
PercentagePhoneCallRatio:DerivedMultiInstanceMeasure
- scale :Real
- unitOfMeasure :-
- function :(TotalPhoneCallReceived / (TotalPhoneCallReceived+TotalEmailReceived)) * 100
isCalculatedUsingPM
isCalculatedUsingPM
TotalPhoneCallReceived:AggregatedMeasure
- scale :Real
- unitOfMeasure :-
- aggregationFunction :sum
- samplingFrequency :-
TotalEmailReceived:AggregatedMeasure
- scale :Real
- unitOfMeasure :-
- aggregationFunction :sum
- samplingFrequency :-
aggregates
aggregates
:CountMeasure
:CountMeasure
meets
meets
:TimeInstantConditionCondition
- stateConsidered :received
:TimeInstantCondition
- stateConsidered :received
appliesTo
appliesTo
PhoneCall:Message
Email:Message

Figure D.8: Incoming customer phone call ratio (Phone Call Ratio).

On the feasibility of measuring performance using PPINOT in CMMN

Monitoring and measuring the performance of business processes are valuable tasks that facilitate the identification of possible improvement areas within the organisation according to the fulfilment of its strategic and business goals. A large number of techniques and tools have been developed with the aim of measuring process performance, but most of those processes are structured processes, usually defined using BPMN. The object of this appendix is to identify and analyse the feasibility of using an existing mechanism for the definition and modelling of process performance indicators (PPINOT) in flexible business process in which the order in which activities are executed is not relevant. This type of processes are usually called unstructured processes [165]. This analysis is based on the similarities between CMMN and BPMN, and on characteristics and attributes used by PPINOT to get values from the business process.

A common example of processes that requires flexibility in their definitions is found in the medical world, where a process represents a medical procedure that may cover a large variety of tasks, but not all tasks are performed by all patients, because each patient has particular needs and represents a different case. In this context, the *Case* concept was introduced [58] and it is defined as *"a proceeding that involves actions taken regarding a subject (a person, a legal action, a business transaction, or some other focal point) in a particular situation to achieve a desired outcome."* [120] and facilitates the specification of processes mostly defined in ad-hoc manners.

As far as we know, there are no formal proposals focused on the performance measuring of unstructured processes. For that reason, we propose to extend a formal technique previously used for the definition of process performance indicators to measure performance over a particular type of unstructured processes. Specifically, we propose to extend the PPINOT Metamodel for defining performance indicators in unstructured processes defined using CMMN [120]. CMMN is a specification that defines a metamodel and notation for modelling and graphically representing a *Case* and a set of common elements used by a *Case*; as well as a format for exchanging case models among different tools. See Section §3.3.1 for more details.

In this appendix, a set of steps is suggested as a working guide to analyse, evaluate and implement possible changes that allow us to measure the performance of CMMN models using PPINOT. This is a preliminary approach that comprises the first steps of our assess, which analyse the feasibility of using PPINOT in the context of CMMN models.

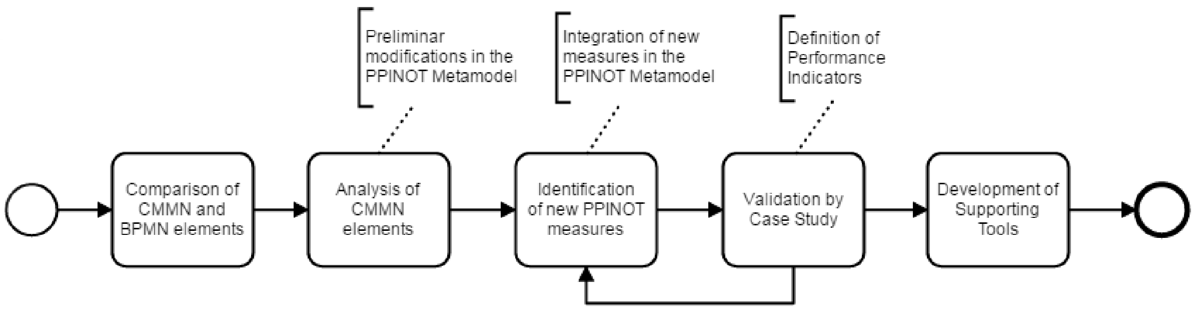

Preliminar modifications in the PPINOT Metamodel
Integration of new measures in the PPINOT Metamodel
Definition of Performance Indicators
Comparison of CMMN and BPMN elements
Analysis of CMMN elements
Identification of new PPINOT measures
Validation by Case Study
Development of Supporting Tools

Figure E.1: Steps of our complete proposal of using *PPINOT* for measuring performance in *CMMN Cases*.

E.1 Proposal for Performance Measurement in CMMN Cases

This section describes five steps identified with the aim of describing how PPINOT can be analysed, evaluated and used for measuring performance over CMMN models. Figure §E.1 shows these steps as activities of a structured process and the purpose of each step is briefly described below.

Comparison *of CMMN and BPMN elements*. PPINOT can be used for measuring performance in business process models regardless of the business process modelling language used to model them, but usually, it is used in BPMN models. This step proposes to identify similarities between the modelling of BPMN elements and CMMN elements as a first integration point of PPINOT.

Analysis *of CMMN elements*. This step seeks to identify characteristics of BPMN elements that PPINOT uses to define "when" and "how" to take values from the business process, and then to identify similar characteristics in the CMMN elements. As a result of this step, preliminary modifications may be included in the PPINOT metamodel.

Identification *of new PPINOT measures*. For those CMMN elements that can not be directly related with a BPMN element, or with a characteristic used for PPINOT; it is necessary to analyse the possibility to measure their performance. In addition, it is necessary to consider PPIs usually defined in CMMN cases to evaluate the need of defining new PPINOT measures.

Validation *by case study*. Once modification points in the metamodel have been identified and changes have been integrated, this proposal should be validated using a case study that includes the definition of a Case and a set of real PPIs defined over it. From this step, the need of including new measures in the metamodel can be identified and those measures should be integrated.

***Development** of Supporting Tools*. After a complete validation cycle of our proposal, a supporting tool should be developed to facilitate the modelling and processing of information taken from the *Case*.

E.2 Analysis of CMMN elements

This section describes a preliminary approach that comprises the first two steps of the process presented in figure §E.1, which are focused on the feasibility of measuring performance of CMMN using PPINOT.

To address the first step, *Comparison between CMMN and BPMN elements*, we analysed functionality and purpose of CMMN elements to identify similarities with BPMN elements. PPINOT was designed regardless of the business process modelling language used to model the process. In previous works, PPINOT was primarily used over BPMN processes, for this reason we consider the identification of similarities as a starting point to establish a link between PPINOT and CMMN.

The second step, *Analysis of CMMN elements*, is based on the way that PPINOT uses to take values from the process, which uses two attributes: *states* of elements and *data*. On the one hand, PPINOT uses `Conditions` to indicate the moment when a measure should take values from the process. Those conditions are based on the different states that a business process element may have during the business process lifecycle. With these states, PPINOT can measure time, counts and states conditions. On the other hand, PPINOT uses `DataContentSelections` to describe a data attribute where the value is taken. Data measure is responsible for measuring this type of values.

Table §E.1 summarizes the results of the two steps. First column shows CMMN elements. This is not a complete list. We based our analysis on CMMN elements associated with the first level of the graphical notation of CMMN. For example, a *CMMN task* has various types: human task, process task, etc., but in this preliminary proposal, this level of detail is not included and only Task element is considered.

Second column represents similarities found between CMMN and BPMN elements. `Case Model` is associated with a *BPMN Pool* because represents the general structure where other elements are contained. `Case File Item` has similar characteristics that a `Data Object` although the first has a broader purpose. `Stage` can be seen as a type of *BPMN Sub-Process* because contains and organises CMMN elements. `Plan Fragment` is similar to a `BPMN Group` because it does not add functionality, it only groups elements.

Types of `Task` in CMMN are different to types of `BPMN Tasks`, but in this level both are considered as the same component; a similar situation applies to `Event Listeners` and *BPMN Events*. `Links` and `Artifacts` are more related to graphical notation. *Links* are similar to `BPMN Flow` and `Artifacts` are similar in both cases because represent `Text annotations`. `Sentry, Milestone` and `Planning Tables` do not have a similar

Table E.1: Table of relationship between CMMN elements, BPMN elements and PPINOT measures

CMMN elements	Similar to BPMN	Has states	PPINOT base measures			
			Time	Count	State	Data
Case Plan Model	Pool	✓	✓	✓	✓	
Case File Item	Data Object	✓	✓	✓	✓	✓
Stage	Sub-Process	✓	✓	✓	✓	
Sentry (criterion)		✱	✓	✓	✓	
Plan Fragment	Group					
Task	Task	✓	✓	✓	✓	
Milestone		✓	✓	✓	✓	
Event Listener	Event	✓	✓	✓	✓	
Planning Table						
Link	Flow					
Artifact	Artifact					

representation in BPMN elements.

Has states column marked with ✓ indicates that the CMMN specification defines states for that element. If this column is marked with ✱ indicates that states are suggested in the specification, but they are not explicitly mentioned. The following elements may adopt different states, reason why PPINOT measures may be applied to measure values over them.

- **Case File Item:** Available, Discarded.
- **Case Instance:** Active, Suspended, Completed, Terminated, Failed, Closed.
- **Stage and Task:** Available, Enabled, Disabled, Active, Suspended, Failed, Completed, Terminated.
- **Event Listener and Milestone:** Available, Suspended, Completed, Terminated

Although `Sentry` specification does not includes Sentry states, the same specification defines it as *a combination of events and/or conditions* and then is possible to say if a Sentry *"is satisfied"* or not. For this reason a `Sentry` is marked with ✱, because we consider a Sentry as a similar element to a BPMN Event. `Plan fragment`, `Planning Table`, `Link` and `Artifact` do not have states related with them.

Finally, the last four columns are related to PPINOT `BaseMeasures`. `Aggregated` and `Derived` measures use `BaseMeasures` to calculate their values. If a `BaseMeasure` can be applied over an element, `Aggregated` and `Derived` measures can be also applied, reason why in this table we only focused on representing the relationship between CMMN elements and `BaseMeasures`.

`TimeMeasure` is used to measure the duration of time between two time instants and `CountMeasure` calculates the number of times that something happens. Each instance is defined by a specific element `state` and the condition to define when something happen is also established by a element `state`. We associated a Time measure with all CMMN elements that have states. We also include a Sentry, because is considered as a similar BPMN event. Similar situation occurs with a `StateConditionMeasure` that evaluates the fulfilment of certain condition in a process instance. In Table §E.1, all CMMN elements that have `states` can be evaluated using a State condition measure. Finally, `DataMeasure` takes a value from a certain part of a data object. In the context of CMMN we could apply this measure to a `Case File Item`, because has similar characteristics to a `DataObject`.

Figure §E.2 shows an excerpt of the PPINOT Metamodel. In this figure, white boxes represent original PPINOT classes; gray boxes represent business process elements, which are connected with original PPINOT classes; and black boxes represent new or modified classes that allow us to include CMMN elements in the PPINOT elements.

According to our preliminary analysis about feasibility of using PPINOT to measure performance of CMMN models, we identify three areas that need to be modified to carry out the integration of the CMMN elements in the PPINOT metamodel. Most changes need to be applied over elements involved in conditions required to specify connections between measures and sources from which the information is obtained. These changes are described below:

- ***Association: relatedTo***. In the original version of the metamodel, a `PPI` is associated with a `BPMN Process`. The name Process remains unchanged, but now it can be instantiated as a *Business Process* or as a *CMNN Case*. This allows the use of CMMN elements in performance measuring.

- ***Association: appliesTo***. In the original version of PPINOT `Condition` class is `appliedTo` a `BPElement`. In our extension, a `Condition` is appliedTo a `SourceElement`. A `SourceElement` can be a `BPElement` (Task, Process, Event or DataObject), or a `CMElement`. A `CMElement` can be a `CasePlan, CasePlanItem, State, PlanFragment, Task, Milestone, Event` or `Sentry`. All these elements are related with *Time, Count* and *State* columns in Table §E.1.

- ***Association: data***. In the original PPINOT version, a `DataContentSelection` class, which is used to indicate the data attribute from which the information is taken, was connected with a `BP DataObject`. In our extension, a `DataContentSelection` is connected with a `Data` class; and a `Data` class can be instantiated as a BP `DataObject` or with a CMMN `CaseFileItem` class.

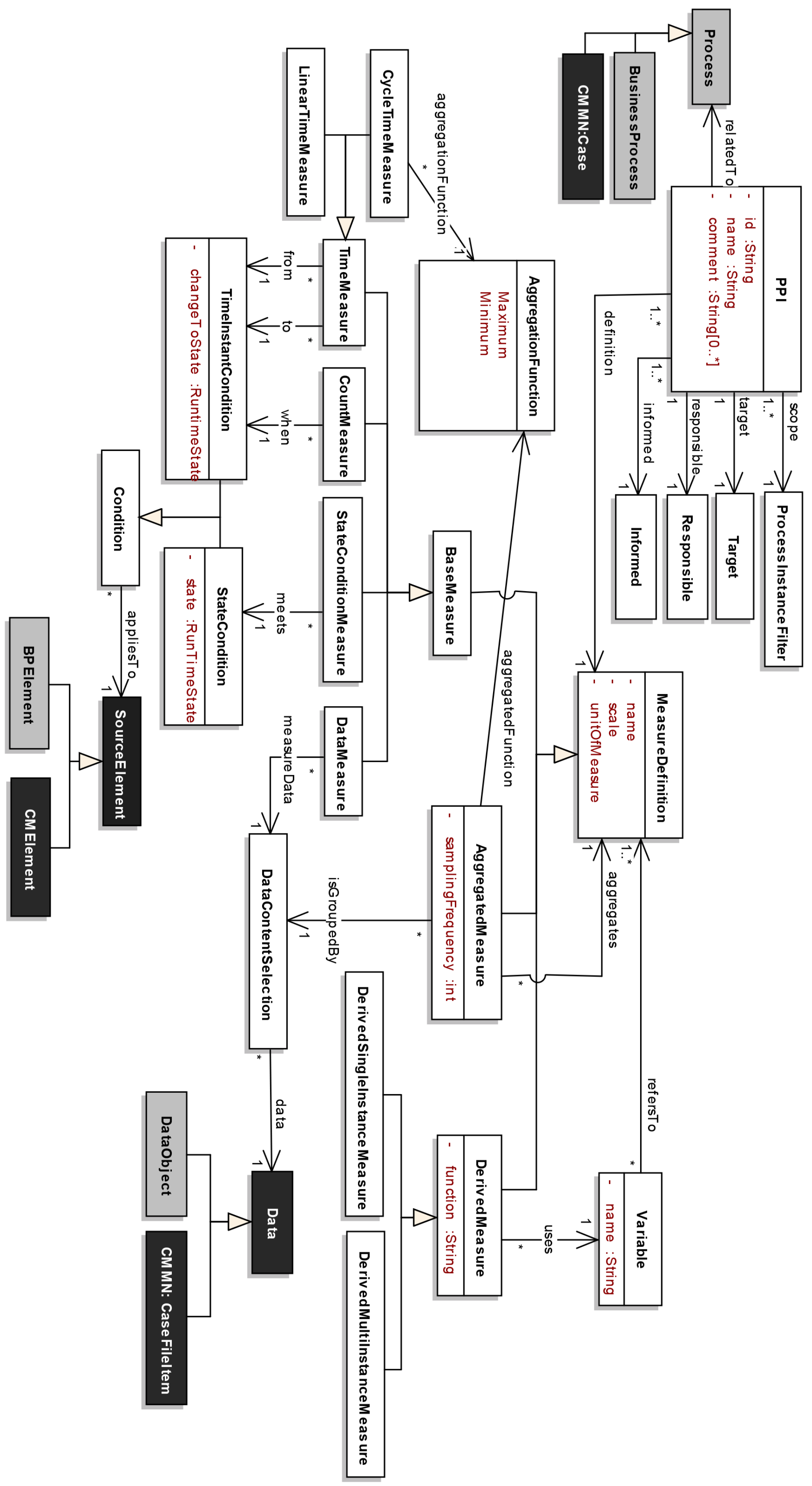


Figure E.2: Excerpt of the PPINOT Metamodel including connections with CMMN elements

E.3 Discussion and Future Work

The analysis of CMMN elements introduced in the previous section comprises the first two steps of a complete analysis, still in progress, that seeks to extend a technique used for the definition of performance indicators over a different context from the business processes.

Although this is not a exhaustive analysis, because we are not including all possible CMMN elements and we are only using CMMN elements included in the CMMN notation, this analysis provides a positive answer to the question about the possibility of using PPINOT elements to measure the performance of CMMN elements.

While it is true that CMMN and BPMN were conceived with different specific purposes, both seek to provide the better way to represent a set of actions related to each other to achieve a goal. Similarities found between them have been the starting point to relate PPINOT with a different context such as CMMN. In addition to similarities in functionality of elements, states found in CMMN, similar to those used in BPMN, facilitate the definition of conditions over CMMN elements that allows PPINOT measures to take values to measure performance.

In order to continue and improve with the evaluation of our proposal, next steps should include the complete set of CMMN elements, not only those that have graphical representation. It is possible that new elements require a deep analysis to identify new ways to get values. This analysis may generate the definition of new PPINOT measures, and therefore the metamodel should change and be extended. The validation by means of case studies based on real scenarios is recommended to recognize real needs about measuring performance. All these actions, including the development of a supporting tool for modelling Cases using CMMN and PPIs using PPINOT, are included in the last three activities of the process described in Section §E.1 that constitute the future work of the current proposal.

Bibliography

[1] M. Acher, P. Collet, P. Lahire, and R. B. France. Managing variability in workflow with feature model composition operators. In *International Conference on Software Composition - SOCO 2010*, pages 17–33, 2010. (page 51).

[2] M. Aiello, P. Bulanov, and H. Groefsema. Requirements and tools for variability management. In *Computer Software and Applications Conference Workshops (COMPSACW)*, pages 245–250, 2010. (page 47).

[3] L. Aksoy, B. Cooil, and N. H. Lurie. Decision quality measures in recommendation agents research. *Journal of Interactive Marketing*, 25(2):110 – 122, 2011. (pages 59, 63, 155, 163).

[4] L. Aldin and S. de Cesare. A literature review on business process modelling: new frontiers of reusability. *Enterprise Information Systems*, 5(3):359–383, 2011. (pages 40, 42, 69).

[5] C. Alexander, S. Ishikawa, and M. Silverstein. *A pattern language: towns, buildings, construction*. Oxford University Press, New York, 1977. (page 42).

[6] S. C. C. APICS. *Supply Chain Operations Reference Model: SCOR Version 11.0*. Supply Chain Council, 2015. APICS, CCOR, CPIM, CSCP, DCOR, SCOR, and SCORmark are all registered trademarks of APICS. All rights reserved. (pages xvi, 12, 68, 95, 179, 180).

[7] APQC, American Productivity & Quality Center. APQC's Process Classification Framework (PCF). `https://www.apqc.org/pcf`, 2018. [Online; accessed 28-June-2018]. (page 179).

[8] Axelos, Global Best Practices. ITIL. `https://www.axelos.com/best-practice-solutions/itil`, 2018. [Online; accessed 28-June-2018]. (page 179).

[9] K. Batoulis, A. Baumgraß, N. Herzberg, and M. Weske. Enabling Dynamic Decision Making in Business Processes with DMN. In *Business Process Management Workshops: BPM 2015*, pages 418–431, 2015. (pages 63, 152).

[10] K. Batoulis, A. Meyer, E. Bazhenova, G. Decker, and M. Weske. Extracting Decision Logic from Process Models. In *CAiSE 2015*, pages 349–366, 2015. (pages 62, 152).

[11] E. Bazhenova and M. Weske. Deriving Decision Models from Process Models by Enhanced Decision Mining. In *Workshops BPM*, pages 444–457, 2016. (pages 59, 63, 156, 163, 164).

[12] A. Bevacqua, M. Carnuccio, F. Folino, M. Guarascio, and L. Pontieri. A data-driven prediction framework for analyzing and monitoring business process performances. In S. Hammoudi, J. Cordeiro, L. A. Maciaszek, and J. Filipe, editors, *Enterprise Information Systems*, pages 100–117, Cham, 2014. Springer International Publishing. ISBN 978-3-319-09492-2. doi: 10.1007/978-3-319-09492-2_7. (page 6).

[13] U. S. Bititci, A. S. Carrie, and L. McDevitt. Integrated performance measurement systems: an audit and development guide. *The TQM Magazine*, 9(1):46–53, 1997. doi: 10.1108/09544789710159443. (page 24).

[14] J. Blasini and S. Leist. Success factors in process performance management. *Business Process Management Journal*, 19(3):477–495, 2013. doi: 10.1108/14637151311319914. (page 25).

[15] R. Bobrik, M. Reichert, and T. Bauer. View-based process visualization. In *International Conference on Business Process Management*, pages 88–95. Springer, 2007. (page 41).

[16] P. C. Brewer and T. W. Speh. Using the balance scorecard to measure supply chain performance. *Journal of Business Logistics*, 21(1):75–93, 2000. (page 26).

[17] J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst. *Mining Configurable Process Models from Collections of Event Logs*, pages 33–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. (page 138).

[18] D. Calvanese, M. Dumas, Ü. Laurson, F. M. Maggi, M. Montali, and I. Teinemaa. Semantics and Analysis of DMN Decision Tables. In *BPM*, pages 217–233, 2016. (pages 59, 63).

[19] Canadian Supply Chain Sector Council. Supply Chain Definitions. `http://www.supplychaincanada.org/en/supply-chain`, 2018. [Online; accessed 28-June-2018]. (page 179).

[20] E. C. S. Cardoso, P. S. Santos Jr., J. P. Andrade Almeida, R. S. S. Guizzardi, and G. Guizzardi. Semantic integration of goal and business process modeling. In *International Conference on Research and Practical Issues of Enterprise Information Systems (CONFENIS 2010)*, 2010. (page 53).

[21] F. T. S. Chan. Performance measurement in a supply chain. *The International Journal of Advanced Manufacturing Technology*, 21(7):534–548, 2003. (page 26).

[22] M.-Y. Chen, M.-J. Huang, and Y.-C. Cheng. Measuring knowledge management performance using a competitive perspective: An empirical study. *Expert Systems with Applications*, 36(4):8449 – 8459, 2009. (page 122).

[23] P. Coad, D. North, and A. Mark Mayfield. *Object Models: Strategies, Patterns, and Applications*. Yourdon Press, 1997. ISBN 0131086146. (page 41).

[24] R. Cognini, F. Corradini, A. Polini, and B. Re. Extending Feature Models to Express Variability in Business Process Models. In *Advanced Information Systems Engineering Workshops - CAiSE 2015 International Workshops, Stockholm, Sweden, June 8-9, 2015, Proceedings - CAiSE Workshop 2015*, volume 215, pages 245–256, 2015. (pages 47, 49).

[25] P. A. da Mota Silveira Neto, I. do Carmo Machado, J. D. McGregor, E. S. de Almeida, and S. R. de Lemos Meira. A systematic mapping study of software product lines testing. *Information and Software Technology*, 53(5):407 – 423, 2011. (page 49).

[26] T. H. Davenport. *Process Innovation: Reengineering Work Through Information Technology*. Harvard Business School Press, Boston, MA, USA, 1993. ISBN 0-87584-366-2. (page 20).

[27] J. de Lara, E. Guerra, and J. S. Cuadrado. Reusable abstractions for modeling languages. *Information Systems*, 38(8):1128–1149, 2013. (pages 41, 69, 72).

[28] G. Decker. *Design and Analysis of Process Choreographies*. PhD thesis, University of Potsdam, June 2009. (pages 21, 22).

[29] F. M. del-Rey-Chamorro, R. Roy, B. van Wegen, and A. Steele. A framework to create key performance indicators for knowledge management solutions. *Journal of Knowledge Management*, 7(2):46–62, 2003. (page 137).

[30] A. del Río-Ortega. *On the Definition and Analysis of Process Performance Indicators*. PhD thesis, Universidad de Sevilla, September 2012. (pages xiii, 27).

[31] A. del Río-Ortega, M. Resinas, C. Cabanillas, and A. Ruiz-Cortés. On the definition and design-time analysis of process performance indicators. *Information Systems*, 38(4):470–490, 2013. ISSN 0306-4379. doi: https://doi.org/10.1016/j.is.2012.11.004. (pages xiii, 4, 8, 26, 31, 32, 39, 70, 100, 122, 128, 133, 136, 150, 157, 163).

[32] A. del Río-Ortega, M. Resinas, A. Durán, and A. Ruiz-Cortés. Using templates and linguistic patterns to define process performance indicators. *Enterprise Information Systems*, 10(2):159–192, 2016. doi: 10.1080/17517575.2013.867543. (pages 4, 26, 28, 30, 78).

[33] A. del Río-Ortega, F. García, M. Resinas, E. Weber, F. Ruiz, and A. Ruiz-Cortés. Enriching decision making with data-based thresholds of process-related kpis. In *Proceedings of Conference on Advanced Information Systems Engineering (CAiSE 2017)*, pages 193–209, 2017. (page 139).

[34] A. del Río-Ortega, M. Resinas, A. Durán, B. Bernárdez, A. Ruiz-Cortés, and M. Toro. Visual PPINOT: A Graphical Notation for Process Performance Indicators. *Business & Information Systems Engineering*, tbd(tbd):tbd, 2017. (pages 26, 28, 30, 108, 122, 138).

[35] C. Di Ciccio, A. Marrella, and A. Russo. Knowledge-intensive processes: Characteristics, requirements and analysis of contemporary approaches. *Journal on Data Semantics*, 4(1):29–57, 2015. (pages 5, 52, 135, 138).

[36] R. M. Dijkman, M. La Rosa, and H. A. Reijers. Managing large collections of business process models-current techniques and challenges. *Computers in Industry*, 63 (2):91–97, 2012. (pages 4, 40, 68).

[37] T. Dirgahayu, D. Quartel, and M. van Sinderen. Development of transformations from business process models to implementations by reuse. Technical Report TR-CTIT-07-82, University of Twente, Centre for Telematics and Information Technology (CTIT), Enschede, the Netherlands, June 2007. (page 42).

[38] D. Dowding and C. Thompson. Measuring the quality of judgement and decision-making in nursing. *Journal of advanced nursing*, 44(1):49–57, 2003. (pages 59, 155, 163).

[39] D. Draheim. Decomposing business processes. In *Business Process Technology: A Unified View on Business Processes, Workflows and Enterprise Applications*, pages 119–160. Springer, 2010. (pages 4, 42, 69).

[40] M. Dumas. Decision Management with DMN in Practice, 2016. Retrieved from: http://bpmtips.com/decision-management-with-dmn-in-practice/#Dumas. (pages 59, 150).

[41] M. Dumas, M. La Rosa, J. Mendling, H. A. Reijers, et al. *Fundamentals of Business Process Management*, volume 1. Springer, 2013. ISBN 978-3-642-33142-8. doi: 10.1007/978-3-642-33143-5. (pages 4, 20, 22, 135, 138, 155, 179).

[42] J. Eder and W. Liebhart. Workflow Recovery. In *International Conference on Cooperative Information Systems (CoopIS'96)*, pages 124–134, 1996. doi: 10.1109/COOPIS.1996.555004. (page 22).

[43] R. Eshuis and P. Grefen. Constructing customized process views. *Data & Knowledge Engineering*, 64(2):419 – 438, 2008. ISSN 0169-023X. doi: https://doi.org/10.1016/j.datak.2007.07.003. (page 41).

[44] B. Estrada-Torres, A. del Río-Ortega, M. Resinas, and A. Ruiz-Cortés. JCIS 2015 - actas de las xi jornadas de ingenierÃa de ciencia e ingenierÃa de servicios. In *Reduciendo la complejidad gráfica de indicadores de procesos de negocio usando abstracción*, page 10, Santander, Spain, Sept. 2015. SISTEDES. URL http://hdl.handle.net/11705/JCIS/2015/008. (page 171).

[45] B. Estrada-Torres, A. del Río-Ortega, M. Resinas, and A. Ruiz-Cortés. Identifying variability in process performance indicators. In *BPM Forum 2016*, pages 91–107, 2016. (pages 95, 172).

[46] B. Estrada-Torres, V. Torres, A. del Río-Ortega, M. Resinas, V. Pelechano, and A. Ruiz-Cortés. JCIS 2016 - actas de las xii jornadas de ingeniería de ciencia e ingeniería de servicios. In *Defining PPIs for Process Variants based on Change Patterns*, page 4, Salamanca, Spain, Sept. 2016. SISTEDES. URL http://hdl.handle.net/11705/JCIS/2016/018. (page 171).

[47] B. Estrada-Torres, A. del Río-Ortega, M. Resinas, and A. Ruiz-Cortés. On the feasibility of measuring performance using PPINOT in CMMN. In *JCIS 2017 - Actas de las XIII Jornadas de Ciencia e Ingeniería de Servicios*, page 10, Tenerife, Spain, jul 2017. SISTEDES. URL http://hdl.handle.net/11705/JCIS/2017/021. (page 172).

[48] B. Estrada-Torres, A. del Río-Ortega, M. Resinas, and A. Ruiz-Cortés. On the relationships between decision management and performance measurement. In J. Krogstie and H. A. Reijers, editors, *Advanced Information Systems Engineering*, pages 311 – 326, Cham, 2018. Springer International Publishing. ISBN 978-3-319-91563-0. (pages 150, 173).

[49] J. D. Evans. *Straightforward Statistics for the Behavioral Sciences*. Brooks/Cole Publishing Company, 1996. (page 143).

[50] M. Fellmann, A. Koschmider, and A. Schoknecht. Analysis of business process model reuse literature: Are research concepts empirically validated? In *Modellierung*, pages 185–192, 2014. (pages 4, 40).

[51] W. Frakes and C. Terry. Software Reuse: Metrics and Models. *ACM Computing Surveys (CSUR)*, 28(2):415–435, 1996. (page 42).

[52] J.-P. Friedenstab, C. Janiesch, M. Matzner, and O. Muller. Extending bpmn for business activity monitoring. In *Hawaii International Conference on System Sciences*, pages 4158–4167, 2012. (pages 26, 28).

[53] B. Ganter and G. Stumme. Creation and merging of ontology top-levels. In *Conceptual Structures for Knowledge Creation and Communication*, pages 131–145, 2003. (page 125).

[54] J. Ghattas, P. Soffer, and M. Peleg. Improving business process decision making based on past experience. *Decision Support Systems*, 59:93 – 107, 2014. (pages 63, 163).

[55] O. González, R. Casallas, and D. Deridder. MMC-BPM: A Domain-Specific Language for Business Processes Analysis. In *Proceedings of Business Information Systems (BIS 2009)*, volume 21, pages 157–168, 2009. (pages 26, 28).

[56] E. Gray and D. Tall. Abstraction as a natural process of mental compression. *Mathematics Education Research Journal*, 19(2):23–40, 2007. (page 41).

[57] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.-C. Shan. Business Process Intelligence. *Computers in Industry*, 53(3):321 – 343, 2004. (page 139).

[58] A. Grudzinska-Kuna. Supporting knowledge workers: case management model and notation (CMMN). *Information Systems in Management*, Vol. 2, No 1, 2013. (page 223).

[59] G. Guizzardi. *Ontological foundations for structural conceptual models*. PhD thesis, University of Twente, 2005. (page 52).

[60] G. Guizzardi, R. de Almeida Falbo, and R. S. Guizzardi. Grounding software domain ontologies in the unified foundational ontology (ufo): The case of the ode software process ontology. In *Iberoamerican Workshop on Requirements Engineering and Software Environments (IDEAS 2008)*, pages 127–140, 2008. (pages 52, 128, 131).

[61] A. Hallerbach, T. Bauer, and M. Reichert. Guaranteeing soundness of configurable process variants in provop. In *IEEE Conference on Commerce and Enterprise Computing 2009*, pages 98–105, 2009. (pages 49, 51, 94).

[62] A. Hallerbach, T. Bauer, and M. Reichert. *Configuration and Management of Process Variants*, pages 237–255. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-00416-2. doi: 10.1007/978-3-642-00416-2_11. (page 46).

[63] A. Hallerbach, T. Bauer, and M. Reichert. Capturing Variability in Business Process Models: The Provop Approach. *Journal of Software: Evolution and Process*, 22 (6_7):519–546, 2010. (pages 49, 51, 94, 95, 107).

[64] M. Hammer and J. Champy. *Reengineering the Corporation: A Manifesto for Business Revolution*. Harper Business, 1993. ISBN 9780887306402. (page 20).

[65] K. Henttonen, A. Kianto, and P. Ritala. Knowledge sharing and individual work performance: an empirical study of a public sector organisation. *Journal of Knowledge Management*, 20(4):749–768, 2016. (page 58).

[66] S. Herbold, J. Grabowski, and S. Waack. Calculation and optimization of thresholds for sets of software metrics. *Empirical Software Engineering*, 16(6):812–841, 2011. (page 139).

[67] H. Heß. *In Corporate Performance Management: ARIS in Practice*, chapter From Corporate Strategy to Process Performance - What Comes after Business Intelligence? Springer-Verlag, Berlin Heidelberg, 2005. (page 25).

[68] O. Holschke, J. Rake, and O. Levina. Granularity as a cognitive factor in the effectiveness of business process model reuse. In *International Conference on Business Process Management - BPM 2009*, pages 245–260, 2009. (page 40).

[69] R. A. Howard and A. E. Abbas. *Foundations of Decision Analysis*. Prentice-Hall, 2015. (pages 59, 155).

[70] S. H. Huan, S. K. Sheoran, and G. Wang. A review and analysis of supply chain operations reference (SCOR) model. *Supply Chain Management: An International Journal*, 9(1):23–29, 2004. (page 70).

[71] R. Hull, A. Koschmider, H. A. Reijers, and W. Wong. Fresh approaches to business process modeling (dagstuhl seminar 16191). *Dagstuhl Reports*, 6(5):1–30, 2016. (page 52).

[72] J. Huysmans, B. Baesens, and J. Vanthienen. A new approach for measuring rule set consistency. *Data & Knowledge Engineering*, 63(1):167 – 182, 2007. (pages 59, 63, 150).

[73] C. Iochpe, C. Ming Chiao, G. Hess, G. Nascimento, L. Thom, and M. Reichert. Towards an intelligent workflow designer based on the reuse of workflow patterns. In *Simpósio Brasileiro de Sistemas Multimídia e Web*, 2007. (page 42).

[74] Ö. Işik, W. Mertens, and J. Van den Bergh. Practices of knowledge intensive process management: quantitative insights. *Business Process Management Journal*, 19(3):515–534, 2013. (page 58).

[75] Ö. Isik, W. Mertens, and J. Van den Bergh. Practices of knowledge intensive process management: quantitative insights. *Business Process Management Journal*, 19(3):515–534, 2013. (page 52).

[76] L. Janssens, E. Bazhenova, J. De Smedt, J. Vanthienen, and M. Denecker. Consistent Integration of Decision (DMN) and Process (BPMN) Models. In *CAiSE Forum*, pages 121–128, 2016. (pages 62, 152).

[77] H. Kaindl, R. Popp, R. Hoch, and C. Zeidler. Reuse vs. reusability of software supporting business processes. In *Software Reuse: Bridging with Social-Awareness: 15th International Conference, ICSR 2016, Limassol, Cyprus, June 5-7, 2016, Proceedings - ICSR 2016*, pages 138–145, 2016. (pages 4, 69).

[78] R. S. Kaplan and D. P. Norton. The balanced scorecard: Measures that drive performance. *Harvard Business Review*, 70(1):71–79, 1992. (page 26).

[79] G. Keren and W. B. De Bruin. On the assessment of decision quality: Considerations regarding utility, conflict and accountability. *Thinking: Psychological perspectives on reasoning, judgment and decision making*, pages 347–363, 2003. (pages 59, 155).

[80] J. Kolb and M. Reichert. A flexible approach for abstracting and personalizing large business process models. *SIGAPP Appl. Comput. Rev.*, 13(1):6–18, 2013. (page 42).

[81] B. Korherr and B. List. Extending the EPC and the BPMN with business process goals and performance measures. In *Proceedings of International Conference on Enterprise Information Systems (ICEIS 2007)*, pages 287–294, 2007. (pages 26, 28, 122).

[82] I. V. Kozlenkova, G. T. M. Hult, D. J. Lund, J. A. Mena, and P. Kekec. The role of marketing channels in supply chain management. *Journal of Retailing*, 91(4”):586 – 609, 2015. ISSN 0022-4359. doi: https://doi.org/10.1016/j.jretai.2015.03.003. Past, Present, and Future of Marketing Channels. (page 179).

[83] E. Krauth, H. Moonen, V. Popova, and M. C. Schut. Performance measurement and control in logistics service providing. In *Proceedings of International Conference on Enterprise Information Systems (ICEIS 2005)*, pages 239–247, 2005. (page 26).

[84] J. Krogstie. *Modelling Languages: Perspectives and Abstraction Mechanisms*, pages 89–204. Springer London, 2012. (pages 41, 69, 72).

[85] A. Kronz. *Managing of Process Key Performance Indicators as Part of the ARIS Methodology*, pages 31–44. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-30787-7. doi: 10.1007/3-540-30787-7_3. (page 27).

[86] C. W. Krueger. Software reuse. *ACM Computing Surveys*, 24(2):131–183, 1992. (pages 4, 69).

[87] P. Kueng. Process performance measurement system: A tool to support process-based organizations. *Total Quality Management*, 11(1):67–85, 2000. (page 122).

[88] M. La Rosa, M. Dumas, A. H. Ter Hofstede, and J. Mendling. Configurable multi-perspective business process models. *Information Systems*, 36(2):313–340, 2011. (pages 7, 48, 51, 94, 95, 112).

[89] M. La Rosa, M. Dumas, R. Uba, and R. Dijkman. Business process model merging: An approach to business process consolidation. *ACM Trans. Softw. Eng. Methodol.*, 22(2):11:1–11:42, Mar. 2013. ISSN 1049-331X. doi: 10.1145/2430545.2430547. (page 48).

[90] M. La Rosa, W. M. Van Der Aalst, M. Dumas, and F. P. Milani. Business process variability modeling: A survey. *ACM Computing Surveys (CSUR)*, 50(1):2, 2017. (pages xiii, 5, 47, 48, 49, 50, 51, 107, 119).

[91] K. C. Lee, S. Lee, and I. W. Kang. KMPI: measuring knowledge management performance. *Information & Management*, 42(3):469–482, 2005. (pages 58, 122).

[92] R. Lenz and M. Reichert. It support for healthcare processes - premises, challenges, perspectives. *Data and Knowledge Engineering*, 61(1):39–58, 2007. (page 148).

[93] T. A. Little and A. V. Deokar. Role of Knowledge Asset Indicators in Understanding Knowledge-Intensive Processes. In *Proceedings of Americas Conference on Information Systems (AMCIS 2011)*, 2011. (page 57).

[94] T. A. Little and A. V. Deokar. Understanding knowledge creation in the context of knowledge-intensive business processes. *Journal of Knowledge Management*, 20 (5):858–879, 2016. (page 52).

[95] A. Lodhi, V. Köppen, S. Wind, G. Saake, and K. Turowski. Business Process Modeling Language for Performance Evaluation. In *47th Hawaii International Conference on System Sciences*, pages 3768–3777, Jan 2014. doi: 10.1109/HICSS.2014.469. (page 94).

[96] I. Machado, R. Bonifácio, V. Alves, L. Turnes, and G. Machado. Managing variability in business processes: An aspect-oriented approach. In *Conference on Aspect-Oriented Software Development - Workshop on Early Aspects*, pages 25–30, 2011. (page 49).

[97] F. M. Maggi, R. P. J. C. Bose, and W. M. P. van der Aalst. Efficient discovery of understandable declarative process models from event logs. In *Proceedings of Conference on Advanced Information Systems Engineering (CAiSE 2012)*, pages 270–285, 2012. (page 138).

[98] O. Marjanovic and R. Freeze. Knowledge intensive business processes: Theoretical foundations and research challenges. In *Proceedings of Hawaii International Conference on Systems Science (HICSS-44 2011*, pages 1–10, 2011. (page 52).

[99] I. Markovic and A. C. Pereira. Towards a formal framework for reuse in business process modeling. In *Business Process Management Workshops: BPM 2007 International Workshops, BPI, BPD, CBP, ProHealth, RefMod, semantics4ws, Brisbane, Australia, September 24, 2007, Revised Selected Papers - BPM Workshops 2007*, pages 484–495, 2008. (pages 4, 40, 41, 68, 69).

[100] F. Martin. A performance technologist's approach to process performance improvement. *Performance Improvement*, 47(2):30–40, 2008. doi: 10.1002/pfi.184. (page 25).

[101] A. P. Massey, M. M. Montoya-Weiss, and T. M. O'Driscoll. Performance-centered design of knowledge-intensive processes. *Journal of Management Information Systems*, 18(4):37–58, 2002. (pages 20, 57, 122).

[102] C. McChesney, S. Covey, and J. Huling. *The 4 disciplines of execution: Achieving your wildly important goals*. Simon and Schuster, 2012. (pages 123, 136, 138, 139).

[103] Meiliana and D. Suhartono. Variability Model Implementation on Key Performance Indicator Application. *International Journal of Innovation Management and Technology*, 6(1):77–80, 2015. (page 51).

[104] Meiliana, M. Vianden, and H. Lichter. Variability model towards a metric specification process. In *International Conference on Computer Science and Information Technology*, pages 76–79, 2011. (page 51).

[105] M. Mikyeong, H. Minwoo, and Y. Keunhyuk. Two-level variability analysis for business process with reusability and extensibility. In *International Computer Software and Applications Conference - COMPSAC 2008*, pages 263–270, 2008. (page 49).

[106] F. Milani, M. Dumas, and R. Matulevičius. Identifying and Classifying Variations in Business Processes. In *BMMDS-EMMSAD - CAiSE 2012*, volume 113, pages 136–150, 2012. ISBN 978-3-642-31071-3. (pages 47, 94).

[107] F. Milani, M. Dumas, N. Ahmed, and R. Matulevičius. Modelling families of business process variants: A decomposition driven method. *Information Systems*, 56:55–72, 2016. (page 94).

[108] C. Momm, R. Malec, and S. Abeck. Towards a model-driven development of monitored processes. In *Proceeding of Internationale Tagung Wirtschaftsinformatik (WI 2007)*, pages 319–336, 2007. (pages 26, 28, 122).

[109] C. Monsalve, A. April, and A. Abran. Business process modeling with levels of abstraction. In *IEEE Colombian Conference on Communication and Computing - IEEE COLCOM 2015*, pages 1–6, 2015. (page 42).

[110] D. Moody. The Physics of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering*, 35(6):756–779, Nov 2009. ISSN 0098-5589. doi: 10.1109/TSE.2009.67. (pages 84, 85).

[111] E. V. Moura, F. M. Santoro, and F. Araujo Baião. Xcutekip: Support for knowledge intensive process activities. In *Collaboration and Technology. CRIWG 2015*, pages 164–180, 2015. (pages 141, 142).

[112] W. Münster. RISE_BPM. Propelling BPM by Research and Innovation Staff Exchange, 2016. Retrieved from: `http://www.rise-bpm.eu/`. (page 13).

[113] J. Nakatumba and W. M. P. van der Aalst. Analyzing resource behavior using process mining. In S. Rinderle-Ma, S. Sadiq, and F. Leymann, editors, *Business Process Management Workshops*, pages 69–80, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-12186-9. doi: 10.1007/978-3-642-12186-9_8. (page 6).

[114] A. Neely, M. Gregory, and K. Platts. Performance measurement system design: a literature review and research agenda. *International journal of operations & production management*, 15(4):80–116, 1995. (pages 24, 25).

[115] A. Neely, H. Richards, J. Mills, K. Platts, and M. Bourne. Designing performance measures: a structured approach. *International Journal of Operations & Production Management*, 17(11):1131–1152, 1997. (page 137).

[116] O. Negulescu and E. Doval. The Quality of Decision Making Process Related to Organizations' Effectiveness. *Procedia Economics and Finance*, 15:858 – 863, 2014. (pages 152, 154, 155).

[117] N. F. Noy and D. L. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical report, Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880, Stanford, CA, 2001. (page 52).

[118] A. A. Nura and N. H. Osman. A toolkit on effective decision making measurement in organizations. *International Journal of Humanities and Social Science*, 2(4): 296–303, 2012. (pages 59, 154, 155).

[119] I. Object Management Group. Business Process Model And Notation (BPMN), Version 2.0, 2011. (pages 22, 28).

[120] I. Object Management Group. Case Management Model and Notation (CMMN), Version 1.0, 2014. (pages 56, 124, 223).

[121] I. Object Management Group. Decision Model And Notation (DMN) V1.1, 2016. (pages xiii, xiv, 59, 60, 62, 153, 156).

[122] I. Object Management Group. Unified Modeling Language (UML), Version 2.5.1, 2017. (page 21).

[123] O. Oyemomi, S. Liu, I. Neaga, and A. Alkhuraiji. How knowledge sharing and business process contribute to organizational performance: Using the fsqca approach. *Journal of Business Research*, 69(11):5222–5227, 2016. (page 58).

[124] M. Paludo, R. Burnett, and E. Jamhour. Patterns leveraging analysis reuse of business processes. In *International Conference on Software Reuse: Advances in Software Reusability - ICSR 2000*, pages 353–368, 2000. (page 41).

[125] C. Pedrinaci, D. Lambert, B. Wetzstein, T. van Lessen, L. Cekov, and M. Dimitrov. Sentinel: A semantic business process monitoring tool. In *Proceedings of International Workshop on Ontology-supported Business Intelligence 2008.*, pages 26–30, 2008. (pages 26, 28, 122).

[126] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3):45–77, 2007. (page 11).

[127] J. M. Perez-Alvarez, M. T. Gomez-Lopez, L. Parody, and R. M. Gasca. Process Instance Query Language to Include Process Performance Indicators in DMN. In *Enterprise Distributed Object Computing Workshop (EDOCW)*, pages 1–8, 2016. (pages 59, 63, 156, 164).

[128] R. G. Poluha. *Application of the SCOR model in supply chain management*. Cambria Press, Youngstown, New York, 2007. (page 70).

[129] A. Polyvyanyy, S. Smirnov, and M. Weske. Process Model Abstraction: A Slider Approach. In *International IEEE Enterprise Distributed Object Computing Conference*, pages 325–331, 2008. (pages 41, 42).

[130] V. Popova and A. Sharpanskykh. Modeling organizational performance indicators. *Information Systems*, 35(4):505–527, 2010. ISSN 0306-4379. doi: https://doi.org/10.1016/j.is.2009.12.001. (pages 4, 26, 28, 78, 122).

[131] Y. Qian, C. Dang, J. Liang, H. Zhang, and J. Ma. On the evaluation of the decision performance of an incomplete decision table. *Data & Knowledge Engineering*, 65 (3):373–400, 2008. (pages 63, 150).

[132] Y. Qian, J. Liang, D. Li, H. Zhang, and C. Dang. Measures for evaluating the decision performance of a decision table in rough set theory. *Information Sciences*, 178(1):181 – 202, 2008. (pages 63, 150).

[133] Y. Qian, J. Liang, P. Song, C. Dang, and W. Wei. Evaluation of the decision performance of the decision rule set from an ordered decision table. *Knowledge-Based Systems*, 36:39 – 50, 2012. (pages 59, 63, 150).

[134] A. Rannacher, R. Stranzenbach, F. Sturm, S. Mütze-Niewöhner, and C. M. Schlick. A system dynamics model for the evaluation of the productivity of knowledge-intensive services. *iBusiness*, 5(3B):55–58, 2013. (page 57).

[135] A. S. Rao and M. P. Georgeff. Bdi agents: From theory to practice. In *Proceedings of International Conference on Multi-Agent Systems (ICMAS-95)*, pages 312–319, 1995. (page 53).

[136] M. Razavian and R. Khosravi. Modeling Variability in Business Process Models Using UML. In *Fifth International Conference on Information Technology: New Generations, 2008. ITNG 2008 - ITNG 2008*, pages 82–87, 2008. (page 51).

[137] M. Reichert and B. Weber. *Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies*. Springer, 2012. ISBN 978-3-642-30408-8. doi: 10.1007/978-3-642-30409-5. (pages 5, 46).

[138] M. Reichert, A. Hallerbach, and T. Bauer. Lifecycle Management of Business Process Variants. In *Handbook on Business Process Management 1*, pages 251–278. Springer Berlin Heidelberg, 2015. (page 47).

[139] H. A. Reijers, R. Mans, and R. A. van der Toorn. Improved model management with aggregated business process models. *Data & Knowledge Engineering*, 68(2): 221–243, 2009. (page 42).

[140] P. J. Richard, T. M. Devinney, G. S. Yip, and G. Johnson. Measuring organizational performance: Towards methodological best practice. *Journal of Management*, 35 (3):718–804, 2009. (page 139).

[141] C. Richter-von Hagen, D. Ratz, and R. Povalej. Towards self-organizing knowledge intensive processes. *Journal of Universal Knowledge Management*, 0(2):148–169, nov 2005. (pages 6, 57).

[142] C. Rolland and S. Nurcan. Business process lines to deal with the variability. In *Hawaii International Conference on System Sciences - HICSS 2010*, pages 1–10, 2010. (pages 49, 94).

[143] M. Rosemann and W. M. P. van der Aalst. A configurable reference modelling language. *Information Systems*, 32(1):1–23, 2007. (pages 48, 51, 112).

[144] J. Saldivar, C. Vairetti, C. Rodríguez, D. Florian, F. Casati, and R. Alarcón. Analysis and improvement of business process models using spreadsheets. *Information Systems*, 57:1 – 19, 2016. (pages 26, 28).

[145] J. B. d. Santos França, J. M. Netto, J. do E.S.Carvalho, F. M. Santoro, F. A. Baião, and M. Pimentel. Kipo: the knowledge-intensive process ontology. *Software & Systems Modeling*, 14(3):1127–1157, 2015. (pages xiii, 5, 52, 53, 54).

[146] A.-W. Scheer, O. Thomas, and O. Adam. Process modeling using event-driven process chains. *Process-aware information systems*, pages 119–146, 2005. (page 21).

[147] A. Schnieders. Variability mechanism centric process family architectures. In *International Conference and Workshop on Engineering of Computer Based Systems (ECBS 2006)*, pages 289–298, 2006. (page 51).

[148] S. Smirnov, H. A. Reijers, and M. Weske. A Semantic Approach for Business Process Model Abstraction. In *Advanced Information Systems Engineering: 23rd International Conference, CAiSE 2011, London, UK, June 20-24, 2011. Proceedings - CAiSE 2011*, pages 497–511, 2011. (pages 4, 42, 69).

[149] S. Smirnov, H. A. Reijers, M. Weske, and T. Nugteren. Business process model abstraction: a definition, catalog, and survey. *Distributed and Parallel Databases*, 30(1):63–99, 2012. (pages 4, 41, 69).

[150] S. Smirnov, M. Weidlich, J. Mendling, and M. Weske. Action patterns in business process model repositories. *Computers in Industry*, 63(2):98–111, 2012. (page 42).

[151] S. Strecker, U. Frank, D. Heise, and H. Kattenstroth. METRICM: a modeling method in support of the reflective design and use of performance measurement systems. *Information Systems and e-Business Management*, 10(2):241–276, 2012. doi: 10.1007/s10257-011-0172-6. (pages 4, 26).

[152] A. Streit, B. Pham, and R. Brown. Visualization support for managing large business process specifications. In W. M. P. van der Aalst, B. Benatallah, F. Casati, and F. Curbera, editors, *Business Process Management*, pages 205–219, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. (page 41).

[153] G. Stumme and A. Maedche. Fca-merge: Bottom-up merging of ontologies. In *IJCAI*, 2001. (page 125).

[154] F. Sturm, A. Rannacher, L. Straub-Bilan, R. Stranzenbach, and S. Mütze-Niewöhner. Productivity measurement of knowledge-intensive services–towards adaptable service performance evaluation systems. In *Proceedings of International conference RESER 2008*, 2011. (page 57).

[155] M. C. Suárez-Figueroa, A. Gómez-Pérez, and M. Fernández-López. *The NeOn Methodology for Ontology Engineering*, pages 9–4. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. (page 125).

[156] L. H. Thom, J. M. Lau, C. Iochpe, and J. Mendling. Extending business process modeling tools with workflow patterns reuse. In *International Conference on Enterprise Information Systems - ICEIS 2007*, 2007. (page 42).

[157] H. Tran, U. Zdun, and S. Dustdar. View-based and model-driven approach for reducing the development complexity in process-driven SOA. In *International Working Conference on Business Process and Services Computing - BPSC 2007*, pages 105–124, 2007. (page 42).

[158] G. Vaidyanathan. A framework for evaluating third-party logistics. *Communications of the ACM*, 48(1):89–94, 2005. (page 26).

[159] W.-J. van den Heuvel. Survey on business process management. Technical report, Tilburg School of Economics and Management, 2008. (page 21).

[160] H. van der Aa, H. Leopold, K. Batoulis, M. Weske, and H. A. Reijers. Integrated Process and Decision Modeling for Data-Driven Processes. In *Business Process Management Workshops: BPM 2015*, pages 405–417, 2015. (pages 62, 152).

[161] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske. Business process management: A survey. In W. M. P. van der Aalst and M. Weske, editors, *Business Process Management*, pages 1–12, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. (pages 20, 23).

[162] W. M. P. van der Aalst, M. La Rosa, and F. M. Santoro. Business process management. *Business & Information Systems Engineering*, 58(1):1–6, Feb 2016. ISSN 1867-0202. doi: 10.1007/s12599-015-0409-x. (page 4).

[163] A. Van Looy and A. Shafagatova. Business process performance measurement: a structured literature review of indicators, measures and metrics. *SpringerPlus*, 5(1):1797, 2016. (page 26).

[164] Z. Wang, P. N. Sharma, and J. Cao. From knowledge sharing to firm performance: A predictive model comparison. *Journal of Business Research*, 69(10):4650–4658, 2016. (page 58).

[165] M. Weske. *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer, 2012. ISBN 978-3-642-28615-5. doi: 10.1007/978-3-642-28616-2. (pages xiii, 4, 20, 21, 22, 23, 25, 223).

[166] B. Wetzstein, Z. Ma, and F. Leymann. Towards measuring key performance indicators of semantic business processes. In *Proceedings of Business Information Systems (BIS 2008)*, pages 227–238, 2008. (pages 26, 28, 122).

[167] A. Yousfi, R. Saidi, and A. K. Dey. Variability patterns for business processes in BPMN. *Information Systems and e-Business Management*, 14(3):443–467, 2016. (page 47).

[168] A. Yousfi, R. Saidi, and A. K. Dey. Variability patterns for business processes in BPMN. *Information Systems and e-Business Management*, 14(3):443–467, 2016. (page 94).

[169] N. Zaaboub Haddar, L. Makni, and H. BenÂ Abdallah. Literature review of reuse in business process modeling. *Software & Systems Modeling*, 13(3):975–989, 2014. (pages 4, 40, 72).

[170] S. Zlatkin and R. Kaschek. Towards Amplifying Business Process Reuse. In *Perspectives in Conceptual Modeling - ER Workshop 2005*, pages 364–374, 2005. (page 41).