



Centro de Informática
Pós-Graduação em Ciências da Computação

“Business Process Configuration with NFRs
and Context-Awareness”

Por

Emanuel Batista dos Santos

Tese de Doutorado



Universidade Federal de Pernambuco

Recife,
May 10, 2013

Emanuel Batista dos Santos

Business Process Configuration with NFRs and
Context-Awareness

Doctoral Thesis

Thesis presented to the Graduate Program in Computer Science of the Universidade Federal de Pernambuco in partial fulfillment of the requirements for the degree of Doctor of Science.

Advisor: Jaelson Freire Brelaz de Castro

Recife,
May 10, 2013

Catálogo na fonte
Bibliotecária Jane Souto Maior, CRB4-571

Santos, Emanuel Batista dos

Business process configuration with NFRs and context-awareness / Emanuel Batista dos Santos. - Recife: O Autor, 2013.

xiii, 176 f. : il., fig., tab.

Orientador: Jaelson Freire Brelaz de Castro.

Tese (doutorado) - Universidade Federal de Pernambuco. CIn, Ciência da Computação, 2013.

Inclui bibliografia e apêndice.

1. Ciência da computação. 2. Engenharia de software. I. Castro, Jaelson Freire Brelaz de (orientador). II. Título.

004

CDD (23. ed.)

MEI2013 – 100

Tese de Doutorado apresentada por **Emanuel Batista dos Santos** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título “**Business Process Configuration with NFRs and Context-Awareness**” orientada pelo Prof. Jaelson Freire Brelaz de Castro e aprovada pela Banca Examinadora formada pelos professores:

Prof. Alexandre Marcos Lins de Vasconcelos
Centro de Informática / UFPE

Prof. Nelson Souto Rosa
Centro de Informática / UFPE

Profa. Maria Lencastre Pinheiro de Menezes Cruz
Departamento de Engenharia da Computação / UPE

Prof. Paulo Cesar Masiero
Instituto de Ciências Matemáticas e de Computação / USP

Prof. Uirá Kulesza
Departamento de Informática e Matemática Aplicada / UFRN

Visto e permitida a impressão.
Recife, 10 de maio de 2013

Profa. Edna Natividade da Silva Barros
Coordenadora da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Abstract

Business process models are an important source of information for the development of information systems. However, changes in the environment can affect the way business processes are performed. In order to obtain models that reflect the changes it is necessary to continuously check between model and reality.

Good business processes need to be up-to-date and automated to represent the organizational environment. Thus, it is of paramount importance to represent adequately variability in Business Process Models. Representing and configuring business processes variability for a specific organization allows the proper execution of processes. In addition, dynamic environment calls for flexible configuration processes that can meet stakeholders' goals. However, selecting a configuration for business process is a challenging activity. Even though current approaches allow the representation of variability of business process models, the selection of business variants in a given context remains a challenging issue.

In this proposal, we advocate the use of Non-Functional Requirements (NFR) and context-awareness information to drive the configuration of process models at run-time. We propose a model-driven business process configuration approach called Business process Variability Configuration with Contexts and NFRs (BVCCoN), approach that keeps the variability representation, configuration knowledge and contextual information separated from the business process models. We illustrate the BVCCoN model and process through examples. The evaluation of our proposal is based on a simulation assessment.

Keywords: Business Process Modeling, Requirements Engineering, Business Process Model Configuration, Model Transformation

Resumo

Modelos de processos de negócios são uma importante fonte de informação para o desenvolvimento de sistemas de informação. No entanto, as mudanças no ambiente podem afetar a forma como os processos de negócios são realizados. A fim de obter modelos que refletem as mudanças é necessário o alinhamento entre o modelo e a realidade.

Processos de negócios bons precisam ser atualizados e automatizados para representar o ambiente organizacional. Assim, é de suma importância representar adequadamente a variabilidade em Modelos de Processo de Negócio. Representar e configurar a variabilidade em processos de negócio de uma organização específica permite a execução adequada dos processos. Além disso, o ambiente dinâmico exige processos de configuração flexíveis que possam atender os objetivos das partes interessadas. No entanto, selecionando uma configuração para o processo de negócio é uma atividade desafiadora. Mesmo que as abordagens atuais permitam a representação da variabilidade dos modelos de processos de negócios, a seleção de variantes de negócios em um determinado contexto continua a ser uma questão a ser explorada.

Nesta proposta, defendemos o uso de Requisitos Não Funcionais (RNF) e informações de contexto para conduzir a configuração de modelos de processos em tempo de execução. Propomos uma abordagem orientada a modelos para a configuração de processos de negócios chamada BVCCoN, a abordagem mantém a representação variabilidade, conhecimentos de configuração e informação contextual separados dos modelos de processos de negócios. A avaliação da proposta baseia-se numa análise de simulação, assim como uma comparação com algumas outras obras relevantes.

Palavras-chave: Modelagem de Processo de Negócio, Engenharia de Requisitos, Configuração de Processo de Negócio, Transformação de Model.

Acknowledgements

I would like to express my gratitude to my supervisor Jaelson Castro for the useful comments and remarks through the learning process. Without his guidance this thesis would not have been possible.

Numerous fellows from the Laboratório de Engenharia de Requisitos (LER) have contributed to this work with their encouragement, criticisms and discussions. I would like to thank my colleagues Tarcisio Pereira, Almir Buarque, Monique Soares and Gabriela Guedes for helping in the thesis assessments. My sincere thanks also goes to João Pimentel, whose collaboration reviewing and discussing this thesis has been invaluable for me.

I would like to thank the professors that helped with comments and criticisms in the beginning of this work – Fernanda Alencar, Juan Sanchez and Oscar Pastor. As well as, I would like to thank my committee members – Maria Lencastre, Uirá Kulesza, Paulo Masiero, Nelson Rosa, and Alexandre Vasconcelos – whose meticulous comments were an enormous help to me.

I would also like to express my gratitude to the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for their financial support.

Last but not least, I would like to thank my parents Deodato and Neves for their endless love and support. As well as, I would like to thank my brother João and my sisters Larissa and Ana for their support during this journey.

Contents

Abstract	i
Resumo	ii
Acknowledgements	iii
Table of Contents	iv
List of Figures	viii
List of Tables	x
List of Listings	xi
Acronyms	xii
1 Introduction	1
1.1 Problem Area	1
1.2 Problem Statement	4
1.3 Objectives and Approach	6
1.4 Assumptions	7
1.5 Methodology	8
1.6 Outline	8
2 Background and Related Work	10
2.1 Business Process Modeling and Management	10
2.1.1 Business Process Modeling Notation	11
2.1.2 Web Services Business Process Execution Language . .	12
2.1.3 Unified Modeling Language	15
2.1.4 Process-Aware Information System - PAIS	15
2.1.5 Workflow Patterns	16
2.2 Variability Management	18

2.2.1	Software Product Lines	18
2.2.2	Feature Diagrams	19
2.2.3	Software Configuration Management	20
2.2.4	Process Family Engineering	20
2.2.5	Business Family Engineering	21
2.2.6	Configurable Process models	22
2.2.7	Goal-Oriented Variability Approaches	22
2.3	Non-Functional Requirements	23
2.3.1	Definitions and terminology	24
2.3.2	NFR Framework	26
2.3.3	Using NFR-Framework in a Socio-Technical System	28
2.3.4	Conflicts in NFR	32
2.3.5	NFR and Business process models	33
2.4	Context-Awareness	35
2.4.1	Basic Concepts	35
2.4.2	Contextual Information in Business Process	39
2.4.3	Context in Business Process Modeling	40
2.5	Comparison to Related Work	42
2.5.1	Methodology	43
2.5.2	Summary of target works	44
2.5.3	Comparison	45
2.5.4	Discussion of Comparison	53
2.6	Summary	54
3	Model and Metamodel for Business Process Models Configuration with NFR and contexts	56
3.1	Abstract Syntax	57
3.1.1	BPMN model	58
3.1.2	Variability	61
3.1.3	Context-Awareness	64
3.1.4	Non-Functional Requirements	65
3.2	Concrete Syntax	70
3.2.1	Variability perspective	71
3.2.2	NFR perspective	76
3.2.3	Context perspective	78
3.3	Summary	80
4	Configuring Business Process Models with NFR and Contexts	81
4.1	BVCCoN Process	82
4.2	Variability Elicitation	84

4.3	Variability Description	87
4.4	Context Analysis	90
4.5	Link variants & NFR	93
4.6	Perform the configuration	96
4.7	Implementation Issues	100
4.7.1	Configuration Algorithm	100
4.7.2	AHP implementation	101
4.7.3	Model transformation	102
4.8	Summary	106
5	Simulation Based Assessment	107
5.1	Background	109
5.1.1	Assessment template	109
5.1.2	Mann-Whitney test	110
5.2	Airport Application Domain	110
5.2.1	Reference Process	111
5.2.2	Variability elicitation	112
5.2.3	Variability Description	112
5.2.4	Contextual Analysis	115
5.2.5	NFR analysis	116
5.3	Assessment Definition	118
5.4	Assessment Planning	119
5.4.1	Context Selection	119
5.4.2	Hypothesis Formulation	119
5.4.3	Variables	121
5.4.4	Assessment Design	122
5.5	Assessment Operation	124
5.5.1	Preparation	124
5.5.2	Simulation Models	126
5.5.3	Execution	127
5.5.4	Data Validation	128
5.6	Results and Analysis	129
5.6.1	Descriptive Analysis	129
5.6.2	Hypothesis Testing	132
5.7	Discussion	135
5.7.1	Validity of study	137
5.8	Summary	138

6	Conclusions	139
6.1	Contributions	139
6.2	Discussion of Decisions and Findings	140
6.3	Limitations	142
6.4	Future Work	143
6.5	Published Material	144
Appendix A	Metamodel with OCL	146
Appendix B	Model Transformation from BVCCoN to BPMN	152
Appendix C	Airport Simulation Scenario - BPMN Models	159
Appendix D	Airport Simulation Scenario - Contexts and NFRs Models	163
References		166

List of Figures

2.1	BPMN core elements	11
2.2	Example of WS-BPEL file.	14
2.3	Workflow patterns for BPMN from Wohed et al. (2005)	17
2.4	Three essential activities for SPL by Northrop (2002)	19
2.5	Catalogues of NFR via Type	27
2.6	Reliability decomposition of Call Method	30
2.7	Operationalization of NFR	31
2.8	Context Information for the Fire Response example	37
2.9	Process representation with goal models by Lapouchnian et al. (2007)	47
2.10	Questionnaire factual relationship proposed by La Rosa (2009)	49
2.11	Contextualized process models proposed by De La Vara et al. (2010a)	52
3.1	BPMN 2.0 metamodel excerpt adapted from Eclipse.org (2011)	60
3.2	Variability Metamodel adapted from Pohl et al. (2005)	62
3.3	Variability perspective of BVCCoN	63
3.4	Context decomposition metamodel based on Ali et al. (2010) .	65
3.5	PredicateDecomposition with OCL	65
3.6	NFR Framework metamodel	67
3.7	NFR perspective of BVCCoN	68
3.8	NFRSoftgoalContribution with OCL	69
3.9	BVCCoN Diagram Interchange metamodel	71
3.10	Concrete Syntax for VariationPoint	73
3.11	Concrete Syntax for Variant	75
3.12	Concrete Syntax for NFR	77
3.13	Concrete Syntax for Context	79
4.1	BVCCoN process overview	83
4.2	Variability Elicitation	84
4.3	Variability description step	88
4.4	Example of reference model with variation points	88

4.5	Example of Variants and Variation points	90
4.6	Context Analysis	90
4.7	Example of context decomposition	93
4.8	NFR analysis	94
4.9	Use of a NFR Catalogue to select variants	95
4.10	Perform Configuration sub-process	97
4.11	Select new configuration algorithm.	98
4.12	Example of process instance, the AmI System performing some tasks.	100
4.13	Model transformation with QVTO.	104
4.14	Example of Variation Point and Process	105
5.1	Schematic View of Simulation	108
5.2	Figure Reference Process	112
5.3	Variation Points in the reference process	114
5.4	Variation Points and variants with business process parts . . .	114
5.5	NFR and contribution analysis	117
5.6	Data distribution for S0 scenario with variation of NFR priority	130
5.7	Data distribution for S1 scenario with variation of contextual information	131
C.1	Reference model	160
C.2	BPMN Process with priority for Performance NFR	160
C.3	BPMN Process with priority for Reliability NFR	160
C.4	BPMN Process with priority for Security NFR	161
C.5	BPMN Process for Without Internet Context	161
C.6	BPMN Process for Without Online Check-in Context	161
C.7	BPMN Process for Security Issue Context	162
C.8	BPMN Process for Small Airport Context	162

List of Tables

2.1	Process model and Variability	47
2.2	Configuration Mechanisms and Guidance	50
2.3	User Intervention	52
4.1	Variants and their relationships with Non-functional require- ments	95
4.2	Mapping from NFRs Contributions to AHP values	102
4.3	AHP Matrix for reliability	103
5.1	Question and answers for Perform Check-in task	113
5.2	Contributions to NFRs	117
5.3	Computation of NFRs' priority over Variants	125
5.4	Simulated time of tasks	127
5.5	Simulated cost of resources	128
5.6	Statistics of execution time variable	133
5.7	Statistics of resource cost variable	133
5.8	Statistics of execution time variable	134
5.9	Statistics of resource cost variable	135
D.1	Relationship between Tasks and Resources	163
D.2	Relationship between Tasks and Active Contexts	164
D.3	Priorities to NFRs	165

Listings

A.1	Variability perspective metamodel	146
A.2	Contexts perspective metamodel	148
A.3	NFR perspective metamodel	150
B.1	Model Transformation with Operational QVT	152

Acronyms

AHP	Analytic Hierarchy Process.	32
BFE	Business Family Engineering.	21
BPD	Business Process Diagram.	33
BPM	Business Process Management.	2, 10
BPMN	Business Process Modeling Notation.	2, 11
BPMNFR	Business Process Modeling with Non-Functional Requirements.	33
FD	Feature Diagram.	19
IS	Information Systems.	2
MCDA	Multi-Criteria Decision Analysis.	32
NFR	Non-Functional Requirement.	6, 8, 24, 26, 29, 33, 82
PAIS	Process-Aware Information System.	3, 11, 15
PFE	Process Family Engineering.	18, 20, 21
RE	Requirement Engineering.	3, 8
SCM	Software Configuration Management.	20
SIG	Softgoal Interdependency Graph.	28, 32
SPEM	Software & Systems Process Engineering Meta-Model.	143
SPL	Software Product Lines.	3
STS	Socio-Technical System.	1, 29

UML Unified Modeling Language. 15

UML ADs UML Activity Diagrams. 11, 15, 143

WS-BPEL Web Services Business Process Execution Language. 12

WSDL Web Services Description Language. 13

Chapter 1

Introduction

1.1 Problem Area

The development of flexible systems that are understandable, reusable and maintainable has become imperative. In the last few years a class of systems called Socio-Technical System (STS) (Maté & Silva 2005) has grown in importance. The STSs are characterized by the presence of human actors as component of the system, that is, the system is composed by software, hardware and human components. These systems are governed by organizational rules and policies. Since human components follow processes in their work, the business processes are part of a system as well.

A process is a group of activities in a particular sequence that produces goods or provides services. It can be seen as an activity or set of activities where there is an entry, processing, and output (White & Miers 2008). The input of a process can be either a tangible asset or an informational resource. The processes are important for organizations, since it is through them that enterprises and companies perform their functions. Hence, all important activities performed in companies are part of some process.

Process models can be applied in several points in an organization (van Der Aalst 2011): to obtain insights about the process; to discuss and document the process; to verify if the process has been performed correctly; to analyze performance and simulate the participation of people; last but not least, to specify and configure information systems.

The importance of considering the business process in an organization has long been recognized as critical by many software developers. Hence, the software must fit the organization expectation and needs (Chang 2006). As a consequence, the use of business process models as the source for software development has grown and several process-based languages have been

proposed (Montero et al. 2008a). Business processes have become an essential element in the development of Information Systems (IS). An information system designed to automate the activities of a business process can better suit the needs of an organization (Dumas et al. 2005).

Business Process Management (BPM) is a set of techniques, metrics, management best practices and software tools that help organizations model, execute, monitor and optimize their work activities (Suleiman et al. 2009). The business process acts as main point to the BPM allowing the integration of processes in the organization rather than simply integrate data and applications. The Business Process Modeling Notation (BPMN) (OMG 2010a) is an example of a language that has become popular among the business analysts. This notation is based on the representation of working process in term of the activities executed and the flow of action and data in the process. The activities and task of a process can be executed by persons or automatically by software.

A dynamic environment consists of changing surroundings in which the agent reside (Wooldridge 2002). So unlike the static case, the agent must adapt to new situations and overcome possibly obstacles (problems). In business process area, the dynamic environments are characterized by constant changes that may affect the way the processes are performed (Rosemann et al. 2008). This is the case for instance in domains such as transportation and logistics, air companies, insurance companies, and so on. In these environments static processes would become obsolete as the process may be impacted by weather changes, competitors' strategy change, or seasonal demand changes. Thus, circumstances that are predictable may be ignored during execution of process due to this limitation.

Hence, in dynamic environments actions should be applied to adapt the organization processes. This calls for flexibility in their definition (Bentellis & Boufaïda 2008). Process improvements and re-engineering are often required (Chang 2006). Moreover, business processes that run in dynamic environments need immediate response to environment changes. In order to prevent process from terminating without meeting their objectives the changes to the process must be performed timely. If the changes are not performed in due time, the business itself may be hurt. As an example, consider a schedule based processes such as transportation and logistics where a simple delay can create larger impacts due to the chain effects.

When considering dynamic environments a class of system may draw attention: the critical systems (Fowler 2004). The critical systems are related with situation where fail in the system may cause disastrous effects. A safety-critical system is a system that poses a physical hazard to human life (e.g., medical instruments, aircrafts, and control systems in chemical plants). An-

other type of critical system is a mission-critical system, which is one where a hazard can degrade or prevent the successful completion of an intended operation (e.g., military systems, satellites, electrical infrastructure). Moreover, there are the business critical systems, where fails may result in high economic losses (e.g., supply chains, banking account). In these kind of systems the flexibility in business processes may be useful since potential losses compensates the investment in deploy the flexibility mechanisms.

In order to deal with process flexibility there are three main aspects to be considered (Aleixo et al. 2012, Bentellis & Boufaïda 2008): selection, adaptation, and modularity. The flexibility by selection is based on formalisms of modeling that take into account the environmental changes without changing the definition of the BP (Bentellis & Boufaïda 2008). It ensures the existence of a number of alternatives of execution in the BP description at design time. The adaptation refers to the definition of the BP without anticipating the capacity of change of the process at its design time. According to Aleixo et al. (2012), an adequate modularization allows isolating the implementation of a specific feature and reducing the overall complexity of the code. They rely on the representation of variability in business process models. Once the variability is represented it becomes possible to select at design time the best alternative that fits the stakeholders need. It is also possible to adapt the changes in the environment at runtime to take in consideration the opportunities that better fit in the new settings.

The Requirement Engineering (RE) field has incorporated business process models as tools for derive software requirements (De La Vara et al. 2010a). For instance, Process-Aware Information System (PAIS) is a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models (Dumas et al. 2005). Current business process management suites such as jBPM ¹ and Activiti ² have integrated the process definition and execution with the development of information systems that support the process.

The adoption of business processes in connection with software development involves some challenges. Since business processes may vary from one organization to another, and even inside the same organization, the creation of business process models requires special attention. Moreover, the domain where the software will run may be so complex that the business processes as well as the software, which automates some activities of process, change often (Yu et al. 2008).

Some areas such as Software Product Lines (SPL) (Bachmann & Clements

¹<http://www.jboss.org/jbpm>

²<http://www.activiti.org/>

2005) and Autonomic Computing (Cetina et al. 2009) have employed variability analysis to deal with the change in the environment and software. SPL proposes to separate the common parts from the variable ones in order to systematically reuse them to compose solutions that better fit the environment.

On the other hand, the Autonomic Computing proposes the use of self-adaptation and self-configuration strategies to deal with variable behaviors both at design and at run-time. Hence, the representation of their variability (Montero et al. 2008b, Schnieders & Puhlmann 2006) is of utter importance.

Moreover, in more dynamic environments, when using business process models as basis for the development of IS, it should also be taken into account the ability to adapt the business processes to changes in the organizational environment (Reichert & Weber 2012).

1.2 Problem Statement

The need for business process flexibility leads to the need for variability description (Schnieders & Puhlmann 2006). As result, several solutions have been proposed by the industry and academy, like Hallerbach et al. (2009), La Rosa (2009), Montero et al. (2008b), Schnieders & Puhlmann (2006). These proposals describe how to structure or modularize variability and how to obtain customized processes by producing new instances or by configuring process models (Gottschalk et al. 2008). The research has evolved in the last years with the development of technology that support decision such as the business process simulation (Wernick & Hall 2007). However, the problem of choosing the most suitable alternative – the so-called process configuration – is not solved yet. During configuration, analysts shall be able to select and perform actions to obtain the fittest process for a given criteria. Among the problems that should be addressed in the configuration there are: the criteria set that will drive the configuration; how the configuration will be performed; and when configuration actions will be applied.

Some approaches (Montero et al. 2008a, Schnieders & Puhlmann 2006) have applied variability analysis to business process models in order to drive the evolution of the business process. However, these approaches have problems to explain how and why a specific process variant is selected. They do not offer the necessary guidance for the business analyst to configure the process. Despite their benefits, the current approaches have presented few advances towards the configuration of business processes models with high level criteria for run-time adaptation.

During the development of this research the following issues in existing

approaches were identified:

1. **Lack of guidance to process configuration.** Most approaches rely on the user expertise to select the configuration of process models such as Montero et al. (2008b), Schnieders & Puhlmann (2006). The user's expertise is important to obtain reliable business process that will create value to the organization. However, without guidance to the user the selection of variants may become a challenging task. Some approaches do not guide the user to obtain a suitable configuration for a given environment. As consequence the users cannot predict the impact of the decisions and the results.
2. **Lack of multi-criteria to configure process models.** Even approaches that use adaptation strategies to provide process configuration require a clear criteria to configure the process. Some approaches use process metrics such as performance and cost to guide the process design (Hallerbach et al. 2009). However, in situations where important attributes such as security and reliability are priority, the use of cost and performance as unique criteria reduces the value of process. In dynamic environments, the process must follow the changes to stay updated. Consequently, a limited set of criteria reduces the utility of the process in dynamic environments. What is required in many situations is a set of multi-criteria to adequately configure the process models.
3. **Strong user's intervention.** In general when the flexibility is achieved through selection there is a need for user's intervention. The strong need for user's intervention requires that the person responsible for carrying out the configuration also to be an expert in both domain and configuration mechanisms. In some circumstances when the changes are required the user may not be present or may not have the necessary expertise to carry out the configuration. Therefore, the user's intervention should be discouraged in the case of process flexibility based on adaptation. Because in the adaptation strategy, processes change at run-time what make the timing important.

A perspective that has emerged is the use of *context-awareness* to deal with changes in business process models (Bessai et al. 2008, De La Vara et al. 2010b, Ploesser, Peleg, Soffer, Rosemann & Recker 2009, Rosemann et al. 2008). A context can be defined as a state of the world that is relevant to an actor goal (Ali et al. 2010) or as a stimulus for change (Rosemann et al. 2008). It allows the identification of variables that impact the structure of business processes. In our work, information about the context can be used to

eliminate invalid solutions. Among the possible solutions described in models there are some alternatives that just make sense in specific contexts. Using models that describe all possible solutions (e.g., all variants in a variability model) lead to incorrectness. In order to keep all variants in a process model, it is necessary to include decisions that access variables that are out of process scope. This practice is recognized as a misuse variability (Rosemann et al. 2008). On the other hand, contexts allow describing the conditions that constraint a model avoiding invalid models.

Non-Functional Requirement (NFR), sometimes called quality requirements, define constraints that the process must comply to. We advocate the use of Non-Functional Requirements (NFRs) as suitable criteria to guide the process configuration. NFRs represent the high-level characteristics from which processes are usually evaluated e.g. cost, performance, accuracy, and the like. Besides, NFRs has been extensively studied (Chung et al. 2000) and the current body of knowledge provides plenty of techniques that can be borrowed and used in this new business process domain. In fact, some recent works (Cardoso et al. 2009, Pavlovski & Zou 2008, Soffer & Wand 2005, Xavier et al. 2010) are already exploring the possibilities of integrating NFR and business process models at design-time. However, they do not apply the NFRs in an adaptation strategy.

Hence, in this thesis we address the following research question: *“How to configure dynamic business process models?”*.

In this thesis we assume the following hypothesis: *“The use of NFRs and contextual information improves the configuration of process models”*.

1.3 Objectives and Approach

We identified the shortcomings of lack of guidance, lack of multi-criteria to configure process models, and strong user’s intervention. In order to face these shortcomings the core objective of this research is to provide guidance to the configuration of business process models, to offer a mechanism that take into account multi-criteria during configuration, as well as a way to reduce user’s intervention during the configuration.

Therefore, we propose the Business process Variability Configuration with Contexts and NFRs (BVCCoN) approach. The BVCCoN is a business process configuration approach that aims to provide support to configuration of business process based on NFRs and contextual information. Our approach covers three perspectives in the business process configuration: the variability description, the Non-Functional Requirements, and the context awareness. The first perspective is concerned with describing the variability of business

process models and the mechanisms necessary to deal with it. In the second perspective we use NFRs to represent qualities of business processes models; it is concerned with the preferences and interference of quality attributes in the process models. The context-awareness perspective incorporates the influences of the environment in the process models. With the association of monitorable information to business process models, we can provide the capability to adapt them to possible environment changes.

The BVCCoN is composed by two parts the BVCCoN model and the BVCCoN process. The model describes the information necessary to perform the configuration including the variability description, the NFR model and the contextual information. The BVCCoN process identifies variation points and variants in a business process model. Moreover, a selection mechanism is used to create new business process models (configurations) with the variants that better suit the system's context and its NFRs. An essential step in this approach is the identification and linking of business process variants with NFRs and contextual variables, which will guide the configuration of the business process.

1.4 Assumptions

Our proposal uses the contextual information to support the evaluation of variants validity. Hence, according to McKinley et al. (2004) our approach may be classified as a parameter adaptation approach since the contexts describe a pre-defined number of solutions. On one hand, by being a parameter adaptation approach, we can guarantee that the adaptation will be performed timely. With that the combination may be tested at design time before be process executed. On the other hand, our approach inherits the limitations of this kind of method such as the limited evolution after the monitoring starts. In order to add a new variant in the approach it is necessary to start the BVCCoN process again.

In this thesis we consider the following assumptions:

- The proposed approach assumes that the process models are running in a Business Process Management System (BPMS) that supports tasks and resources monitoring
- The context monitoring is supported by tools able to collect information of several sources (e.g., sensors, user profiles, BPMS resources)
- We assume that the user of our approach (i.e., the business analyst) has enough knowledge about the business process that is target of analysis

- The analyst has access to information resources enough to elicit and validate the NFRs and contextual information
- We also assume that the conditions captured during context analysis are sufficient to allow adaptation, thus, it is not able to react to unpredictable events

1.5 Methodology

In order to identify the problem area and state the research question we performed an *ad hoc* literature review with results present in Chapter 2. The literature review presented in this thesis does not follow a rigorous literature review report as a systematic literature review (Kitchenham et al. 2007). Instead we performed an *ad hoc* literature review of the main topics. However, a systematic literature review study (Pereira et al. 2013) performed by independent researchers showed that this thesis was still original in their proposal by the time of its defense. Based on the shortcomings identified in the literature review we designed the BVCCoN approach. As part of the approach definition, we demonstrated the technical feasibility of the proposed approach by implementing parts of them as proof of concepts.

The assessment of our approach was based on a simulation exercise. It was designed using a controlled environment. The use of simulation study allows obtaining a quantitative evaluation of our approach pointing out its benefits and limitations.

1.6 Outline

The rest of the thesis is organized as follows:

- **Chapter 2 - Background and Related Work** We present a summary of the most relevant works related to business process modeling, variability analysis for business process modeling and NFR for business process modeling. In order to investigate the relationship between the areas of Business Process Modeling and RE we also present a literature review of relevant works that have been developed in these areas. Moreover, we introduce the NFR basic concepts, the Business Process Modeling Languages, and the concepts about context awareness.
- **Chapter 3 - Model and Metamodel for Business Process Models Configuration with NFR and Contexts** Presents the BVC-

CoN Model to support our approach. This chapter details the integration of models based on metamodels and describe abstract and concrete syntax for BVCCoN models.

- **Chapter 4 - Configuring Business Process Models with NFR and Contexts** Outlines our new approach with a running example. In this chapter we detail the BVCCoN process step by step describing how to build-up the BVCCoN model and how to derive new BPMN models.
- **Chapter 5 - Simulation Based Assessment** This chapter presents an assessment using simulation methods. In the assessment we use Airport check-in and boarding process as a basis to compare our approach against a reference process.
- **Chapter 6 - Conclusion** We present the conclusions and discuss future works of this thesis.
- **Appendix A - Metamodel with OCL** This appendix presents the metamodel described in Chapter 3 with OCL constraints.
- **Appendix B - Model Transformation from BVCCoN to BPMN** Presents the full model transformation presented in Chapter 4.
- **Appendix C - Airport Simulation Scenario - BPMN Models** This appendix contains all BPMN models used in the simulation study presented in Chapter 5.
- **Appendix D - Airport Simulation Scenario - Contexts and NFRs Models** Last but not least, this appendix presents the tables used to describe the simulation scenarios in Chapter 5.

Chapter 2

Background and Related Work

In this chapter we present relevant background information as well as the results of a literature review on Business Process Modeling languages, Non-Functional Requirements and Context-awareness. Moreover, we discuss the management of variability and variability configuration.

2.1 Business Process Modeling and Management

Business Process Management (BPM) is a systematic and structured approach to analyze, improve, control and manage processes with the aim of improving the quality of products and services (Chang 2006). BPM has as goal to improve product and services through a structured approach that centers on systematic design and management of the companies processes (Suleiman et al. 2009).

According to Chang (2006), BPM is based on the following principles:

- Business processes are organizational assets that are central for creating value for customers,
- By measuring, monitoring, controlling, and analyzing business processes a company can deliver consistent value to customers and has basis for process improvement,
- Business processes should be continuously improved. Both incremental (e.g., Six-sigma) and more radical (e.g., Business Process Reengineering) methodologies have been proposed to support process improvement,

- Information technology is an essential enabler for BPM.

Since modeling business process modeling has become a common practice in the organizational environment, a considerable number of modeling languages have been proposed. This section outlines some of the most important modeling languages used to model business process as well as Process-Aware Information System (PAIS) and some workflow patterns.

2.1.1 Business Process Modeling Notation

The Business Process Modeling Notation (BPMN) (OMG 2010a) is an effort of modeling tools vendors to standardize the process Modeling notations. BPMN is a workflow-based notation that is derived from other workflow notations such as the UML Activity Diagrams (UML ADs). The BPMN is based on the flow of activities and artifacts in the process.

Figure 2.1 shows the core set of BPMN elements used to represent a business process model. Basically, this core set is composed of the following elements:

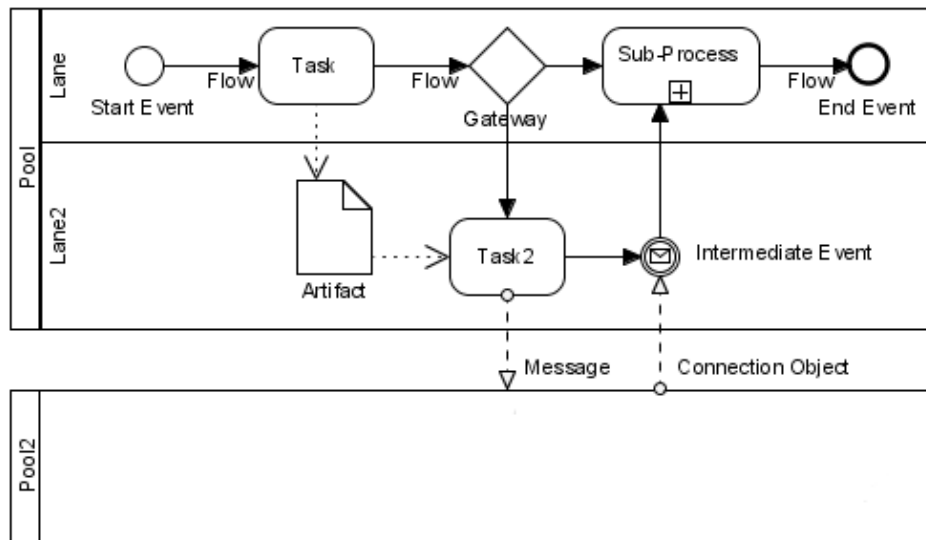


Figure 2.1: BPMN core elements

Event Elements represent the events that start, interrupt, and end a process. The event elements can include several modifiers to indicate special conditions or states of execution;

Activities represent the activities performed in a process. The activities can be high-level activities represented as sub-processes or atomic activities represented as tasks;

Decision elements are the points where the workflow can be split or merged. They are called Gateways and can represent inclusive and exclusive split as well as merge in the process flow. They contain modifiers that indicate the type of gateway.

Connectors link process elements. The sequence flow connectors link elements of flow such as activities, events and gateways. They indicate the sequence of process execution. The message flow connectors represent the message exchange between participants of process; both pools and activities can send and receive messages. The association connectors are used to associate the artifacts with flow elements;

Partition elements are the elements that structure BPMN diagrams. They are called swim-lanes and represent the participants of process. There are two types of swim-lanes: the pools that represented high-level participants of process (e.g., organizations, clients, suppliers), and the lanes that represent the roles or structure of the organization (e.g., Employee, Manager, Sales);

Artifacts are external elements that complement the representation of process. The most important artifacts are the data objects that represent the resources consumed and produced in the process. Other artifacts such as text annotations and groups can be used to give additional information without modifying how the process is performed.

The current version of BPMN includes four Diagrams (OMG 2010a): process diagram, collaboration diagram, choreography diagram, and conversation diagram. They describe the process in several views: the process itself (process diagram), the interaction between participants of process (collaboration diagram), the low-level communication between participants (choreography diagram) and the high-level communication between participants (conversation diagram).

2.1.2 Web Services Business Process Execution Language

The Web Services Business Process Execution Language (WS-BPEL) is a business process modeling language that allows the construction of executable

business processes (Jordan & Evdemon 2007). WS-BPEL represents a convergence between Web services and business process technology. WS-BPEL essentially extends imperative programming languages (e.g. C) with constructs for the implementation of Web Services.

A WS-BPEL process is a hierarchical structure of basic activities corresponding to atomic actions for sending, receiving and creating/processing messages. The language allows the representation of an executable business process with activities, events and resources. It is layered on the top of the WSDL specification since a BPEL process is exposed as a Web Service through Web Services Description Language (WSDL) interfaces.

Figure 2.2 shows an example of WS-BPEL language, the process has a signature (line 1) that indicates association with other elements. In the WS-BPEL process a client, who triggers the process, and the software systems are seen as Web Services. The software systems, also known as partners (line 2) are the concrete realization of the process. The process is executed in a compliant engine that supports the infrastructure of communication between client and its partners. The information produced and consumed is represented by input and output variables (lines 6-8). The process flow is contained in the sequence tag (lines 9-12). In this place several types of steps may appear in the sequence tag, for example, the receive step (line 10) is responsible by receive input from a client.

```

1 <bpel:process name="Fireexample" targetNamespace="Fire"
  suppressJoinFailure="yes" xmlns:tns="Fire" xmlns:bpel="
  http://docs.oasis-open.org/wsbpel/2.0/process/abstract">
2   <bpel:partnerLinks>
3     <bpel:partnerLink name="client" partnerLinkType="tns:
      Fireexample" myRole="FireexampleProvider"
      partnerRole="FireexampleRequester" />
4   </bpel:partnerLinks>
5   <bpel:variables>
6     <bpel:variable name="input" messageType="tns:
      FireexampleRequestMessage"/>
7     <bpel:variable name="output" messageType="tns:
      FireexampleResponseMessage"/>
8   </bpel:variables>
9   <bpel:sequence name="main">
10    <bpel:receive name="receiveInput" partnerLink="client"
      portType="tns:Fireexample" operation="initiate"
      variable="input" createInstance="yes"/>
11    <bpel:invoke name="callbackClient" partnerLink="
      client" portType="tns:FireexampleCallback"
      operation="onResult" inputVariable="output"/>
12  </bpel:sequence>
13 </bpel:process>

```

Figure 2.2: Example of WS-BPEL file.

2.1.3 Unified Modeling Language

The Unified Modeling Language (UML) is a standardized set of modeling languages that offer support for the field of software engineering. Among 14 types of diagrams there is the UML ADs (OMG 2009). The UML ADs are graphical representations of workflows of activities and actions. Primarily designed to model software systems, they also may represent organizational processes such as business process models. They support a variety of activity constructors such as choice, iteration and concurrency. In UML ADs a business process can be described by an activity consisting of a coordinated sequencing of nodes, based on control-flow and object-flow (Russell et al. 2006).

A worthwhile feature of UML is the possibility to extend its vocabulary by defining profiles (La Rosa 2009), which are sets of stereotypes. Stereotypes define new elements from existing ones (e.g. an activity), by adding new properties that are suitable for a specific context.

2.1.4 Process-Aware Information System - PAIS

A Process-Aware Information System (PAIS) has been broadly defined as a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models (Dumas et al. 2005). PAIS is based on the explicit representations of process models. They are the basis for describing the logical and temporal order in which organizational tasks have to be performed (Becker et al. 2003). Moreover, they provide a means of communication between members of organization which are responsible to define business process (e.g., managers, business analysts) and implement the technical infrastructure (e.g., software developers).

According to Dumas et al. (2005) the project of PAIS is organized in three stages:

- The process design where high-level business processes are modeled for scoping the project and capturing and discussing business requirements and process improvement initiatives with domain experts, such as business representatives (La Rosa 2009).
- In the second stage takes place the implementation of process models. The process models that need to be automated are selected and refined into operational (i.e. machine-readable) process specifications, linking the various process tasks to concrete applications and organizational entities (La Rosa 2009).

- The third stage (execution) regards the enactment of executable process specifications. Once a process specification has been deployed to a management system, instances can be triggered by the occurrence of a given event (e.g. the arrival of a purchase order). As a result, one or several tasks belonging to the process instance are enabled and dispatched to relevant people/applications that may perform them, until the whole instance is completed. Finally, in the last stage (diagnosis), operational processes are analyzed to identify problems and to find aspects that can be improved (La Rosa 2009).

2.1.5 Workflow Patterns

The business process modeling languages serve different purposes such as be descriptive for learning, for process development and design, for process execution as well as for Information Technology support (Aguilar-Savén 2004). The modeling languages that are based on workflow, such as BPMN, present a well-defined set of patterns that may be used to structure the model (van Der Aalst et al. 2003, Wohed et al. 2005). These patterns describe how the elements of language can be grouped according to expected behaviors.

Basic patterns such as sequence, parallel split, synchronization, exclusive choice, and Simple merge, cover the elementary aspects of process control. Associated with advanced branching and synchronization patterns it is possible to model logical operations such as AND (sequence, parallel split), XOR (exclusive choice), and OR (Multiple choice, discriminator). Thus, the workflow patterns allow describing mandatory and optional behavior using elements of process language. Figure 2.3 presents the basic workflow patterns according to Wohed et al. (2005). The lines represent the workflow patterns: parallel split, synchronization, exclusive choice, and merge. The columns represent the way how workflow patterns may be represented in BPMN. Observe the same patterns may have several representations.

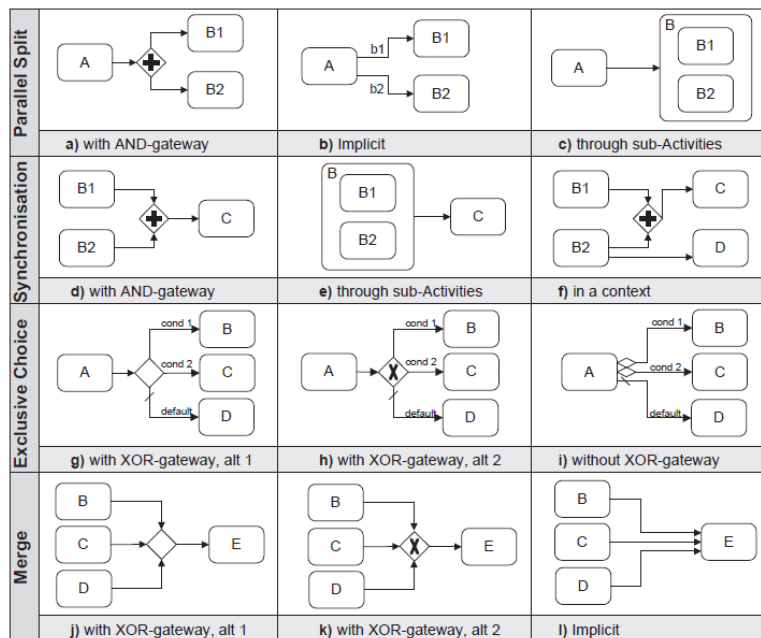


Figure 2.3: Workflow patterns for BPMN from Wohed et al. (2005)

2.2 Variability Management

This Section presents topics related to variability management and some approaches that apply it to the Business Process Modeling activity.

We begin reviewing some key concepts present in software product lines paradigm and one of its basic modeling artifacts the so called feature diagram. When then briefly introduce software configuration management. We also discuss an approach that defines mechanisms for process models, namely the Process Family Engineering (PFE). We then review another work that extends PFE to capture business process variability. In the sequel we comment on approach (La Rosa 2009) that relies on configurable models to keep information about variability. Last but not least we analyze some approaches that rely on goal models to represent variability.

2.2.1 Software Product Lines

In this thesis we share the definition proposed by Bachmann & Clements (2005), which characterizes a Software Product Line a set of software-intensive systems sharing a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets (a core asset is an artifact used in the production of more than one product in a SPL) in a prescribed way.

At its essence, a product line involves core asset development (also known as Domain Engineering) and product development (also known as Application Engineering) using the core assets, both under the supervision of technical and organizational management. Core asset development and product development from the core assets can occur in either order: new products are built from core assets, or core assets are extracted from existing products. Often, products and core assets are built in synergy with each other. Figure 2.4 illustrates this triad of essential activities (Bachmann & Clements 2005).

Each rotating circle represents one of the essential activities. All three are linked together and in perpetual motion, showing that all three are essential, are inextricably linked. They can occur in any order and are highly iterative. The rotating arrows indicate not only that core assets are used to develop products, but also that revisions of existing core assets or even new core assets might, and most often do, evolve out of product development. The diagram in Figure 2.4 is neutral with regards to which part of the effort is launched first. In some contexts, already existing products are mined for generic assets (perhaps a requirements specification, an architecture, or software components) which are then migrated into the core asset base of the product line. In other cases, the core assets may be developed or procured

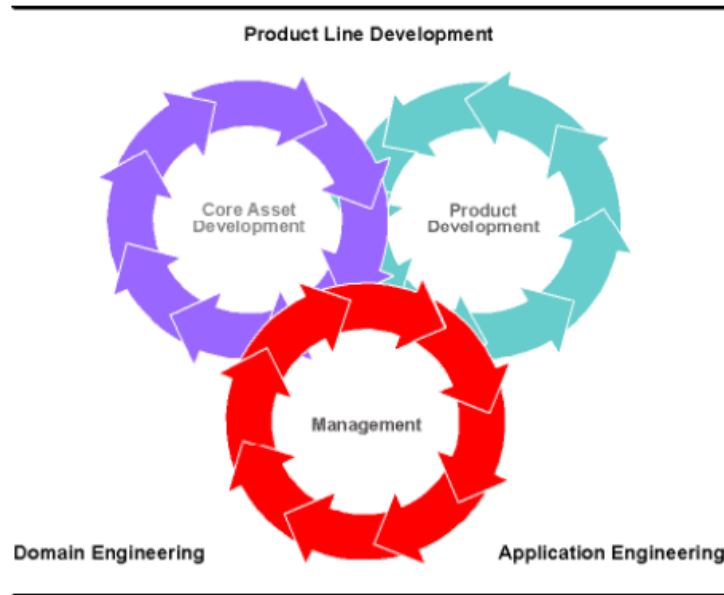


Figure 2.4: Three essential activities for SPL by Northrop (2002)

for later use in the products development.

2.2.2 Feature Diagrams

A feature model is a model that defines features and their dependencies, typically in the form of a feature diagram and left-over (a.k.a. cross-tree) constraints. But it could also be describe as a table of possible combinations. A feature model consists of a tree with the root features, and leafs that decomposes the root features in sub-features. The sub-features decomposed can be: *Mandatory*, when they are required by the root one; *Optional*, when they are not required but can be present; *Alternative*, when just one of the features can be selected; as well as *Or*, when one or more features can be selected. Other decompositions and mechanisms also are allowed in feature models such as the cardinality of features and the exclusion of features.

Feature Diagram (FD) are a family of techniques for describing software product lines in terms of their features. A number of feature modeling languages have been proposed, e.g. Batory & Geraci (1997), Czarnecki & Eisenacker (2000), Czarnecki et al. (2005), Mannion (2002). Note that FDs were first introduced as part of the FODA (Feature Oriented Domain Analysis) method (Kang et al. 1990). A feature model consists of one or more feature diagrams which are generally represented as tree-structures with high-level features being decomposed into sub-features. A feature represents a system

property that is relevant to a stakeholder and is used to capture commonalities or to discriminate among systems in a family (Czarnecki & Eisenecker 2000).

Constraints can be expressed as plain text (Batory & Geraci 1997) or as arbitrary propositional logic expressions over the values of features (Batory 2005), specified by means of a proper grammar (e.g. a limit in the number of sub-features a feature can have). Constraints among the sub-features of a same feature can be graphically represented to model restrictions in the number of sub-features a feature can have. These relations can be: AND (all the sub-features must be selected), XOR (only one sub-feature can be selected) and OR (one or more can be selected). OR relationships can be further specified with an n:m cardinality (Czarnecki et al. 2005), where n indicates the minimum and m indicates the maximum number of allowed sub-features.

2.2.3 Software Configuration Management

The Software Configuration Management (SCM) is a discipline of software development that comprises factors such as configuration identification, configuration control, status accounting, review, building management, process management and team working (Berczuk & Appleton 2002). It works by capturing how a set of available options impacts upon the way a software/hardware system is built from a set of components (Krikhaar et al. 2009).

As a consequence of the growth of software complexity the SCM developed methodologies and environments to support the control of versions and software lifecycle to ensure consistent creation of a baseline software product. These environments keep the information about the variability using version control systems. They apply techniques of branching and merging to generate products from it variants, that is, maintain the commonalities as the stable version and evolve the variants in the unstable versions during development.

2.2.4 Process Family Engineering

The Process Family Engineering (PFE) (Bayer et al. 2005, Schnieders & Puhlmann 2006) is the result of the project PESOA that defined variability mechanisms to process based models including the state chart diagrams, activity diagrams and BPMN models. They investigated the existing variability mechanisms and adapted some of them to be used with the process-based languages. Furthermore, the PFE also defines a method to introduce variability in process-based models.

The PFE process includes the activities of *domain scoping*, *domain analysis*, *application analysis*, *domain design*, *application design*, *domain implementation*, and *application implementation*. The *domain scoping* has the purpose to determine the appropriate bounds of process family. The *domain analysis* defines the features related with the domain to produce a feature model that represents the domain. The *application analysis* activity uses the feature model previously defined to determine what the products that will be built are. In parallel, the *domain design* activity produces a variability-rich process that can be configured to result in specific processes. The *application design* and *domain implementation* activities use the variability-rich process to configure the implementation and then produces software that meets the specified process.

The PFE approach adopts the feature models to analyze variability and commonalities in the process. The variability identified are represented in a process through a set of variability mechanisms (Puhlmann et al. 2005). They are extensions of the process language that allows to represent different mechanisms such as Encapsulation, Parameterization, Inheritance, and Extension Points (Bayer et al. 2005).

2.2.5 Business Family Engineering

The Business Family Engineering (BFE) approach proposed by Montero (2009) describes a methodology that derives systems from business process models applying variability mechanism to obtain reusable process models. The Business Family Engineering starts with the *Business Family Domain Engineering* where the Business Process is designed to be part of the groups: the core processes that are common in this family domain; and the variable processes that can vary in the family domain. The next activity is *Business Family Application Engineering* that uses composition mechanisms to assemble a new process based on core and variable parts. However, the *Business Family Application Engineering* is not detailed in Montero (2009).

The BFE (Montero 2009, Montero et al. 2007, 2008a,b) describes variability mechanisms to represent variability in BPMN models by extending the mechanisms presented in the PFE (Bayer et al. 2005). The Domain Engineering phase is parted in three activities: the *Domain Requirements Engineering*, the *Domain Design* and the *Domain Implementation*. The former uses several requirement engineering techniques to identify and represent domain requirements. The *Domain Design* explores the domain variability and provides a summary of variability and a core architecture to the system. Finally, the *Domain Implementation* is focused on defining the implementation and test details in the architectural core structure. However, this

activity is not detailed in the studies.

2.2.6 Configurable Process models

The flexibility of business process models can also be handled using configurable models (Gottschalk et al. 2008). These type of model are extensions of business process modeling languages that keep information about the variability in the same models that are been modeled.

La Rosa et al. (2009) use a questionnaire-based approach to deal with variability and to drive the configuration of business process models. They model variability in questionnaires and create intermediary models to link them to process models. In their approach, answers to questionnaires offer ways to select among configurations of a process, described in configurable models, i.e., C-EPC or C-YAWL (La Rosa 2009). After the configuration, the business process model can be individualized to obtain a single model without configuration marks.

2.2.7 Goal-Oriented Variability Approaches

Lapouchnian et al. (2007) describe an approach used to design business process using high-variability goal models. It consists in building a goal tree that represents a business process. The goal tree is enriched to support the constructions of business processes that are not supported by the goal notation such as an exclusive OR, conditional events, and sequential AND-decomposition. As result the goal tree can be mapped to the WS-BPEL.

In order to obtain the process, a high-variability model is created, with the most relevant alternatives to run the process. The non-functional requirements are represented by softgoals. They receive contribution of the goals that are variations in the goal tree. These softgoals are the selection criteria to choose an configuration. For instance, if a goal contributes positively to *cost* but negatively to *performance* it can be present in a configuration that prioritize *cost* instead of *performance*.

Liaskos et al. (2006) presents a goal-oriented method to variability analysis. This method uses linguistic frames to identify and analyze variability and commonalities in goal models. The goals are modeled through the construction of hierarchies of AND/OR decompositions of high level stakeholder goals into sub-goals and then, into low-level sub-goals and tasks that lead to requirements of the system-to-be. In the process of analysis the goals are argued in order to identify possible variations in the behavior of the goal. The linguistic frames that make part of the goals decomposition can be the agentive (Who does the action?), the dative (Who is affected by the

action?), the temporal (When the action occurs?), the process (How the action is performed?), and others. Based on this analysis a tree of goals is constructed where each frame can corresponds to a set of possible goal variations of the original one. The idea is to combine the resulting goals into a possible solution. They also propose the reasoning about goals to deal with the selection criteria and the explosive number of solutions. The goals are analyzed in terms of contribution to each other, and then bottom-up or top-down strategies are used to prune the goal tree and reduce the number of possible solutions.

Yu et al. (2008) propose a process that generates a high variability software design from a goal model. This process is supported by heuristic rules that guide the design. The goal models are represented as a goal tree with AND/OR decomposition, the variation points are located in the OR-Decomposition. Moreover, the authors demonstrate that the constructions of a goal tree can be semantically equivalent to the Feature model ones, because for each Feature model construction (e.g. Mandatory, Optional, Or, Alternative) an equivalent goal decomposition can be mapped without loss of information. The goal models can be mapped to three traditional views the feature models (configuration view), the state-chart model (behavioral view), and the architectural component-connector model (structural view). They also present a discussion about mapping the goal model to Business process models.

In González-Baixauli et al. (2007) an extension is presented to deal with the complexity of variability models using an aspect-oriented strategy. The approach supports goal-oriented elements such as goals, softgoal and task, and their decomposition. They allow the goal analysis, by defining correlation and contribution relationships. The goals are related with softgoals and tasks, the alternative way to achieve a goal are represented by task and the selection criteria to choose through analysis of softgoals. During the analysis the authors employ the concepts of separation of concern and crosscutting concerns to separating them in different models. These concepts were borrowed from aspect oriented software development terminology and are applied to join tasks and softgoals that are related to the same concern in the same model. They also argues the using this mechanisms the can better deal with the increasing complexity of the models.

2.3 Non-Functional Requirements

This section describes the Non-Functional Requirement - NFR, their terminology and classification. We also present approaches to handle NFR, and

discuss how NFR, conflicts could be handled. We conclude reviewing some work that related NFRs with business process models.

2.3.1 Definitions and terminology

The Non-Functional Requirement (NFR) are usually associated to quality attributes (Boehm et al. 1978), extra-functional requirements and non-behavioral requirements. On the other hand, the Functional requirements describe the functions of a system, i.e., what the system must do. These requirements will be implemented in software and become available for the users. For this reason, they will have a localized and specific effect. The complexity of a software system is partially determined by functionalities (functional requirements) that represent what the system must do, and by the qualities and constraints associated to the software system (Non-Functional Requirements).

NFR in general are related to quality patterns such as reliability, performance, etc. These requirements are important, because they define if a software will be or not efficient for the task that it is designed to do. According to Kirner & Davis (1996), the NFRs represent additional requirements that defined as global qualities or quality attributes of the system-to-be. Moreover, the NFRs do not express functions to be expressed by the software, but the behaviors and restrictions that the software must achieve (Cysneiros & Leite 1998). Accessibility, security, performance, portability, consistency, maintainability, efficacy, are some examples of NFRs.

The NFRs play a critical role during the development of software systems. They can be the criteria to select among alternatives in design and implementation (Chung et al. 2000). It is well known that errors and omissions in the NFRs can lead to a negative impact over the quality of the final product. The correction of NFRs problems in systems can increase the costs of even make the software unviable. Thus, errors due to missing or incorrect elicitation make then expensive and hard to correct once the system had been implemented (Davis 1993). Conflicts can happen among NFRs or among NFRs and functional requirements (Cysneiros & Leite 1998). As consequence, the lack of attention paid to NFRs can lead to conflicts during the software implementation.

As stated by Chung et al. (2000), there is no formal definition neither a complete list of NFRs. NFRs may vary according to the needs of stakeholders or their perception about the software. For instance, the NFR may be: loosely detailed; technically or economically unviable; interrelated; conflicting; and, confused with functional requirements. In addition there are few methods that deal with non-functional requirements effectively. The main

reason for this is because they are different, domain dependent, and difficult to be specified.

Moreover, there is no unique and universal classification schema that covers all needs of different domain application. As a result, each application domain requires its own classification. For instance, the standard IEEEStd-830 (1998) proposed a list with thirteen (13) NFRs. Other classification of NFRs can be found in Sommerville & Sawyer (1997), which classify the NFR as:

- Process NFR specify constraints for the software development process. Process-oriented approaches integrate the effort to describe and answer NFR during the development process;
- Product NFR specify qualities that the product (i.e., the software) must achieve. Product-oriented approaches assess the degree to which the final product meets certain NFRs;
- External NFRs are derived from the environment where the system will run. They include legal, economic and interoperability restrictions.

In a product-oriented approach the system is measured by the extent that it meets determined NFR through a quality control process capable of doing such checks. Product-oriented approaches are usually employed when the requirements are well defined and can be specified in terms of features and measurable qualitative factors (metrics).

The process-oriented approaches are generally recommended to deal with the NFR in the early stages of requirements analysis.

It is claimed that techniques based on goals are more suitable to represent NFR (Chung et al. 2000). Because they allow to examine the utility of requirements, use a vertical view from strategic level to fine grain level, as well as, use logical operators And/Or, which helps the decision making. The NFR Framework proposed by Chung and colleagues is a widely known approach that could be used for all types of NFR, since the early stages of software development (Chung et al. 2000).

A literature review of NFR from different views was presented in Mairiza et al. (2010). Their research is based on three dimensions: definition and terminology; types; and application domain. The definition and terminology describe how the literature defines NFR and what are the terms used to do so. The type dimension classifies the NFR in categories and identifies the most cited NFR. The third dimension identifies the most important NFR according to the type of system and the application domain.

Following the classification of Mairiza et al. (2010) in next section we present a description of the most cited NFR: Performance, Reliability, Security and Maintainability.

2.3.2 NFR Framework

The NFR Framework (Chung et al. 2000) is a goal-oriented approach designed to be intuitive. It was based on AND-OR trees used in problems solving. However, goals which represent non-functional requirements can rarely be regarded as fully satisfied. Hence the NFRs are treated as softgoals to be achieved, that is they are goals which need to be clarified, disambiguated, prioritized, elaborated upon, etc. Different design decisions contribute positively or negatively towards a particular goal. These goals can be assessed to determine whether a set of non-functional requirements is being achieved by a specific project.

This framework provides a structure to represent and record the design and reasoning processes on graphs, called Softgoal Interdependency Graph (SIG). In addition, it also offers a catalogue of knowledge, including technical development. The NFR-Framework can be used to define non-functional requirements such as performance, reliability, security and maintainability (see Figure 2.5).

Once defined the important NFR for this domain we begin to decompose them. Decomposition can be driven by type, by topic or a specific method (Chung et al. 2000). We use decomposition by type based on the general and specific literature (Chung et al. 2000, Gregoriades et al. 2003, Le 2009). First we produce catalogues with the type decomposition for the NFR. In Figures 2.5 we present catalogues for type decomposition of Maintainability (Fig. 2.5(a)), Performance (Fig. 2.5(b)), Security (Fig. 2.5(c)) and Reliability (Fig. 2.5(d)). The Maintainability is decomposed in the NFRs that are part of its concept according Chung et al. (2000). It is composed by the Understandability, Testability, Modifiability, Changeability and Stability NFRs (Fig. 2.5(a)). Performance (Fig. 2.5(b)) is decomposed in Space Performance and Time Performance, they are two typical aspects of Performance that may be evaluated. Security (Fig. 2.5(c)) is decomposed in Availability, Access Control, Integrity and Confidentiality. They are NFRs that can be further decomposed. Observe that in addition to Fault Tolerance, Consistency and Recoverability, the decomposition of Reliability (Fig. 2.5(d)) reuses the Availability and Integrity NFRs from the Security decomposition.

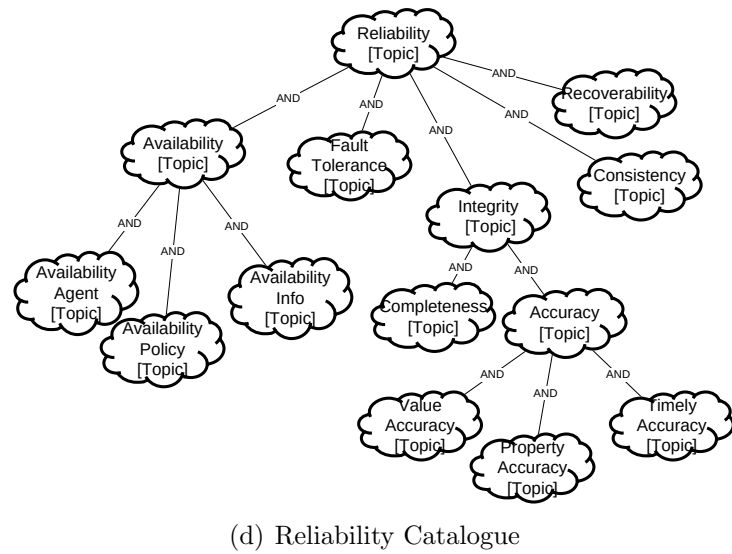
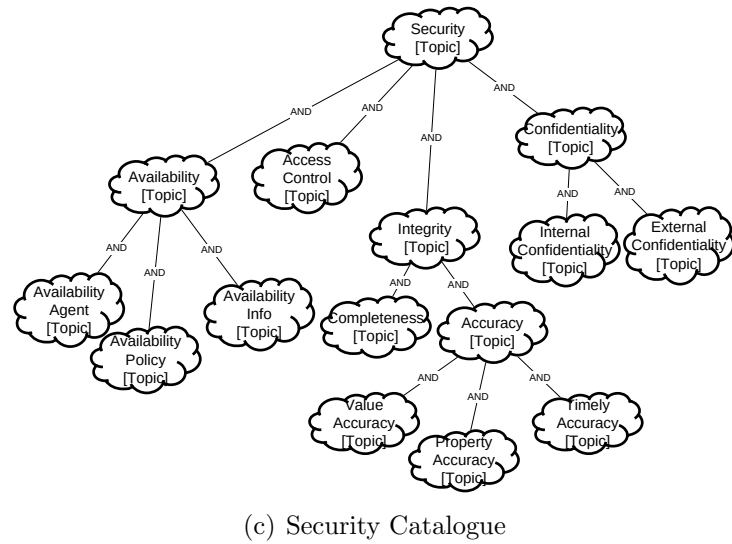
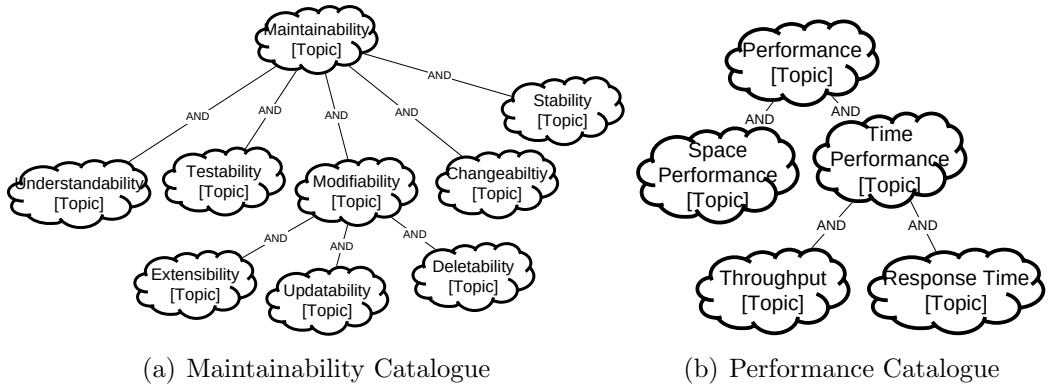


Figure 2.5: Catalogues of NFR via Type

According to Chung et al. (2000), there are several steps in this design process that will be described below.

Acquiring Knowledge The analyst must acquire knowledge about the area (or domain) of the system in question. It includes the functional requirements of the system and specific types of requirements associated with functional and technical development;

Identifying NFR for a specific domain Using the knowledge previously acquired, the analyst shall identify what are the important requirements for the domain, and document it in a Softgoal Interdependency Graph (SIG);

NFR decomposition The decomposition of non-functional requirements is divided into type and topic. The topic decomposition deals with the structural decomposition of the problem, while the type decomposition breakdown a non-functional requirement in its sub-types;

Identifying Operationalization Then the analyst should develop alternatives to meet the non-functional requirements the system in question;

Ambiguity, prioritization and interdependency The SIG may grow and become large and complex. Therefore it is necessary to deal with ambiguities, priorities, and interdependence of operationalization;

Support rationale The Claim-Softgoal is a type of softgoal used to argue about the choices and rationalizes tradeoff;

Selecting among alternatives The decomposition process is performed until the analyst considers those operationalizations are enough to describe the alternatives for the problem. Then the operationalizations can be marked as accepted or denied;

Evaluating the impact of decisions Using bottom-up and top-down analysis is possible to evaluate the impact of the choices, and identify if the operationalizations will be able to achieve softgoals that are associated to them.

2.3.3 Using NFR-Framework in a Socio-Technical System

In this section we will present the use of NFR Framework in a specific domain of an emergency response to a fire where a Socio-Technical System (STS) may be deployed.

The first step is to acquire knowledge about a fire response system. Socio-Technical System (STS) are characterized by the interplay between social and technical components. In STS, the human actors are components of the system along with the software and hardware. Due to the presence of human actors, the STSs present specific properties such as dynamic organizational objectives and non-determinism.

The scenario of crisis management introduced by Dalpiaz et al. (2008) illustrates these properties for Ambient Intelligence (AmI) socio-technical system. When a disaster (e.g., fires, floods, etc.) occurs several responses can be triggered by an AmI. The response to disaster may vary in many points according the type and severity of disaster. In this scenario the processes to deal with the crises also varies since they need to be suitable to a specific situation. Thus, context-awareness and adaptability of processes are essential to this type of process. In this thesis, we will use this scenario as a running example to explain the steps of our approach.

The AmI consists of a monitoring system that will alert if a possible crisis situation arises. It may also identify alternative measures to mitigate or respond to the crisis. Both human and automatic component are involved in the process of identification and response. For example, in the case of fires, the system's smoke detection mechanisms will identify if it is a real menace of fire and activate the alert system (e.g., alarms or call firefighters) accordingly.

The humans may act according to a well-defined process specially defined to response to this type of crisis. The AmI will have to change the process according to the context.

The second step consists of identifying the NFR related with the domain of our problem. According to Mairiza et al. (2010), the NFR most cited in literature for this domain are Performance, Reliability, Security and Maintainability. Other NFR could be identified, but we will focus initially on these ones. The description and relationship of them with STS is provided below:

Security Issues that concern about preventing unauthorized access to the system. The security of the STS (its ability to resist attack) is a complex property that cannot be easily measured. Attacks may be devised that were not anticipated by the system designers and so may defeat built-in safeguards.

Reliability Requirements that specify the capability of software product to operates without failure and maintains a specific level of performance when used under specified normal conditions. In STS, the System reliability depends on component reliability but unexpected interactions

can cause new types of failures and therefore affect the reliability of the system;

Maintainability It describes the capability of the system to be modified that may include correcting a defect or making an improvement or change. This property reflects how easy on hand it is to fix a problem with the system once it has been discovered. It depends on being able to diagnose the problem, access the components that are faulty, and modify or replace these components;

Performance Requirements that specify the capability of software product to provide appropriate performance relative to the amount of resources needed to perform full functionality under stated conditions. The performance of a STS depends not just of the software and hardware performance but also of the operator performance.

Another way to decompose a NFR is using the topic decomposition. Hence, a concept hierarchy is used to decompose the NFR in sub-NFR. Each element of the hierarchy is evaluated for a fixed NFR type. In the Figure 2.6 we present a hierarchy for the concept Call Method, where there are three possible methods to call security services: via internet (automatic alarm), by mobile phone and by land phone line.

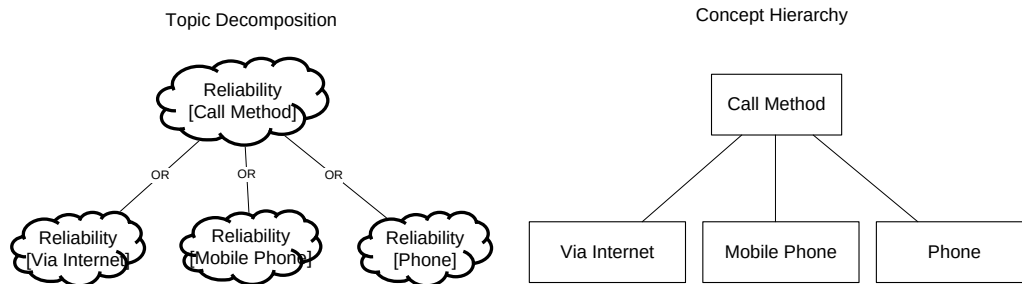


Figure 2.6: Reliability decomposition of Call Method

Once the NFR were decomposed the operationalizations can be linked to the NFR. For example in Figure 2.7, the Response Time[Door opening] NFR can be operationalized by a manual opening or automatic opening. Considering that the doors are locked and that it is necessary to unlock them to leave the building the response time in this situation is essential. If the door just can be opened manually, it requires that the person that has the ability (or authorization) to open the door to be present. Thus, a manual

opening contributes negatively to the response time. On the other hand, if the door can be opened automatically once the fire event is detected it can contribute positively to the response time.

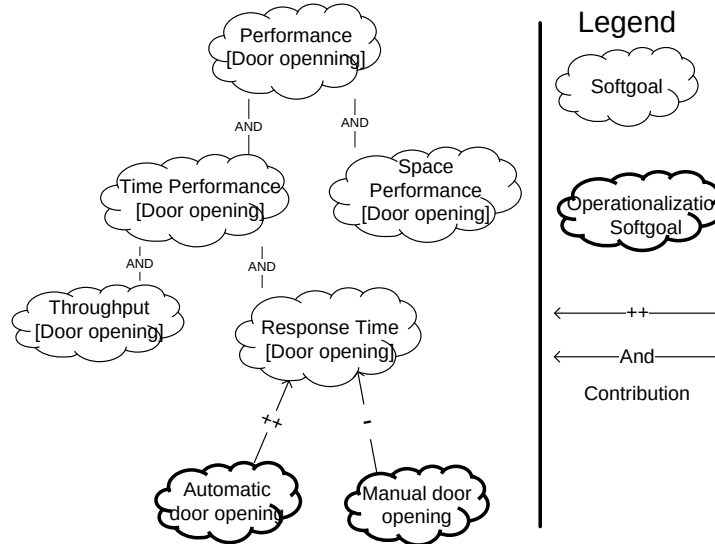


Figure 2.7: Operationalization of NFR

After describing the decomposition of the NFR we start to relate the NFR from different SIGs to obtain the relationships among them. The operationalization identified can be related through correlation to NFR of a different tree. In doing so, it allows to identify conflicts, priorities and so on. For instance, the mechanism to automatically open the doors can be useful to improve the response time but can impact negatively the access control (during the event some doors can be accessed without permission). In order to simplify the visualization the SIG and the contributions and correlations can be translated to tables such as the correlation catalogues (Chung et al. 2000).

The analyst can select the operationalizations that seem to be interesting for the solution and then evaluate the impact of them to the NFR. Using the information of contributions and correlation it is possible to verify whether or not a high level NFR will be achieved. The evaluation algorithms can identify also the presence of conflict in the SIG (Chung et al. 2000).

In the example of STS for a fire event scenario the operationalizations open doors automatically, trigger alarm automatically and call firefighter automatically could be selected to achieve the performance NFR, but the correlations could be evaluated to verify if other NFR are achieved in the

same way. For instance, with the same selection of operationalization Security and Reliability can only be partially satisfied.

2.3.4 Conflicts in NFR

The NFRs may present conflict among them. Sometimes they are implicit such as the conflict among security and performance. To deal with possible conflicts the NFR based approaches present some strategies at design time. The NFR Framework (Chung et al. 2000) uses a propagation algorithm to identify conflicts, thus allowing dealing with this problems at design time. Another available mechanism present in the NFR Framework is the use of prioritization to analyze critical requirements.

The i^* framework (Yu 1995) also deals with NFRs, which are represented as softgoals. Interactive algorithms were proposed to identify the satisfiability of the Softgoals in i^* models (Horkoff & Yu 2010). These algorithms allow the selection of solutions based on qualitative analysis, they rely on the analyst expertise to solve possible conflicts that emerges.

AHP as a conflict solver

Another way to deal with conflicts in the selection of alternatives is the use of Multi-Criteria Decision Analysis (MCDA). These kind of method allow the selection among alternatives based in more than one criteria (Figueira et al. 2005). The MDCA methods deal with multiple and potentially conflicting options. Some methods can deal with qualitative or intangible criteria such as the fuzzy logic or the Analytic Hierarchy Process (AHP) (Saaty 2008). If we consider the NFR as selection criteria, we could apply these methods to deal with conflicts during the selection of alternative options. For instance, the AHP method use pair wise comparisons along prioritization to deal with measurement of multiple qualitative attributes.

As presented in Liaskos et al. (2012) the AHP method may be used to express stakeholders' preferences over NFRs. The AHP calculates the global preference over a criteria by comparing the pair of alternatives. Each pair will provide the preference over a criteria. After compute all alternatives for all criteria the result will be a set of matrices that may be computed for a decision. In the case of NFR Framework, the hierarchical structure of SIG graphs allow the integration with the hierarchical structure of AHP. The use of AHP also allows transforming the qualitative reasoning of NFR Framework into a computable set of values. Moreover, the AHP may be used in several domains as demonstrated by Saaty (2005).

Among the limitations of AHP method we highlight the number of criteria and the explanation of results. The former is the caused by the effect of cumulative calculation of matrices. When all pair comparison matrices are defined they need to be computed in an unique value, during this process some error may stack-up. In order to avoid this problem the author suggest limiting the number of matrices to seven, thus, the quality of solution will not degenerate. The latter limitation of AHP is the lake of results explanation. The result of using AHP is a vector with the order of preference over the alternatives. However, the interpretation of the result is not obvious. Since the results depend on the assessment of all criteria and priority, establish the direct link between the preferences (input) and the ranking (output) is almost impossible. Even so, some experiments (Liaskos et al. 2012) verified that participants generally recognize the result of their preference input and preferred choice.

2.3.5 NFR and Business process models

The importance of NFR for business process models has been acknowledged in the last years. They are applied in the design of business process models through extensions of business process modeling languages, allowing for richer analysis of the process model.

For instance, Xavier et al. (2010) and Pavlovski & Zou (2008) include the NFR into the business process modeling activity. The former by using NFR catalogues during the design and the latter by extending the business process models to incorporate NFR. Neither considers the variability in their solutions.

Xavier et al. (2010) propose an extension to BPMN in order to include non-functional requirements, it is called Business Process Modeling with Non-Functional Requirements (BPMNFR). The BPMNFR adopts the concepts of NFR from the NFR Framework (Chung et al. 2000) to improve the business process models by including the operationalizations of NFR in the model. The BPMNFR approach is composed by the following steps:

1. Create up a Business Process Diagram (BPD),
2. Insert labels in the activities of BPD that represent NFR related with the activities,
3. To use, to extend, or create NFR catalogues,
4. Insert the operationalizations of NFR in the BPD, grouped by the label.

Hence, the BPMNFR allows the creation and refinement a business process model to support the non-functional requirements. However, the BPMNFR needs to maintain the BPD and the NFR models in parallel. Based on the analysis of the BPD, non-functional requirements that are potentially relevant are selected to be detailed (e.g., usability). A NFR catalogue is used (or created) to identify how the activities of the BPD can be modified to support the NFR selected before. The operationalizations of NFR are inserted in the BPD as task or sub-process after being selected.

Pavlovski & Zou (2008) propose an extension to BPMN language to represent non-functional requirements through two new artifacts: *operating condition* and *control case*. They extend the BPMN to allow the representation of concerns and non-functional requirements that are related with the business. The aim of their approach is to identify the non-functional requirements from the business restrictions of a business process model. In their process the business process is modeled and the non-functional requirements are included in the models. Their proposal complements the model without changing the semantic flow and syntactic notation. The identification of requirements is done in a high abstraction level and there is no refinement of requirements. However, their approach does not support the identification of the requirements, neither describes how to detail them. The *operating condition* artifact represents the grouping of constraints that are potential non-functional requirements to be identified. Thus, the *operating condition* is used to declare additional semantic constraints to the process activity. The *control case* artifact uses graphical notation and textual descriptions to represent additional properties of the business constraint the controls to be applied. The *control case* is associated with the *operating condition* that capture the business risk associated with the *operating condition*.

Soffer & Wand (2005) also advocate the importance of NFR (expressed by Softgoals) for the business process design. They provide a conceptual framework to clarify the notion of Softgoal and use it to represent quality in business process models.

Lapouchnian et al. (2007) also describe how to use goal models and annotation mechanisms to derive business process models. They use NFR, represented by Softgoals, to drive the configuration of process. They start their approach with the goal model and then derive the business process model.

2.4 Context-Awareness

The need for systems that dynamically adapt themselves is dramatically increasing. Contextualization can be used as a way to allow adaptation to changes both at design time and at run-time (Rosemann et al. 2008).

The context-awareness has emerged from the demand of applications capable of operating in high dynamic environments such as the pervasive and ubiquitous systems. The context-aware applications aim to meet requirement by taking in consideration the context of use including location, time and activities of the user (Henricksen & Indulska 2004).

In order to perform context-based adaptation, two mechanisms are necessary: monitoring and actuation. The former identifies what should be changed, while the latter performs the changes themselves. These mechanisms are the basis for self-adaptive and autonomic systems (Cetina et al. 2009). The monitoring mechanism requires the definition of attributes to be monitored. This set of attributes is part of the context which can be modeled using contextual analysis.

In this section we describe the basic concepts in context-awareness and the application of context in business process modeling.

2.4.1 Basic Concepts

A context can be defined as a state of the world that is relevant to an actor goal (Ali et al. 2010), or as a stimulus for change (Rosemann et al. 2008). A context can also be defined as the current state of an actor in relation with word objects (Bai et al. 2008). In this thesis we will use the definition proposed by Ali et al. (2010).

The contextualization of models consists in including useful information about the contexts that can affect the model. In the last years the need for systems that are able to dynamically adapt themselves has become a hot research topic. Systems and the design methods must be flexible enough to support changes. In order to do so, contextualization is used as a way to allow adaptation to changes both at design time and run-time.

There are several approaches to model context information (Henricksen & Indulska 2004). In general the context information associated with sensors is stored in repositories. They can be expressed as graphical models or through well-formed expression using relational languages or logic (Bai et al. 2008, Henricksen & Indulska 2004).

The contextual information can be derived from several sources Bettini et al. (2010), including:

Sensorial information Sensors respond to stimulus of the environment generating data. This type of information can be obtained from several sensorial equipment such as GPS, smoke detectors, and so on.

Profiled information In some situation profiles can be used to store contextual information. They can be filled by user (e.g., personal information in webmail accounts) or collected in databases (e.g., number of visited pages in a research engine).

User inputs When users interact with a context-aware system the user inputs can trigger changes.

Context Representation Languages

Several ways to represent the Contextual information have emerged. In the sequel, we present the taxonomy proposed by Bettini et al. (2010) to classify the context representation languages.

Fact-based modeling approaches. They were originated from attempts to create sufficiently formal models of context to support query processing and reasoning, as well as to provide modeling constructs suitable for use in software engineering tasks such as analysis and design (Bettini et al. 2010). Early context modeling approaches, such as attribute-value pairs, could not satisfy these requirements, particularly as the types of context information used by applications became more sophisticated.

Contextual analysis is based on context annotations. Annotations are attached to elements of a model in order to indicate what the relevant context information that can affect that part of the model is. Hence, contextual parts of that model can be enabled or disabled. During the analysis, the contexts can be associated to facts and statements. On one hand facts may be directly assessed, on the other hand statements must be decomposed as facts (Ali et al. 2010). Contexts are linked to sets of facts that can be assessed to identify the validity of the context. In Figure 2.8 we show the context modeling of our running example, i.e. the fire response STS.

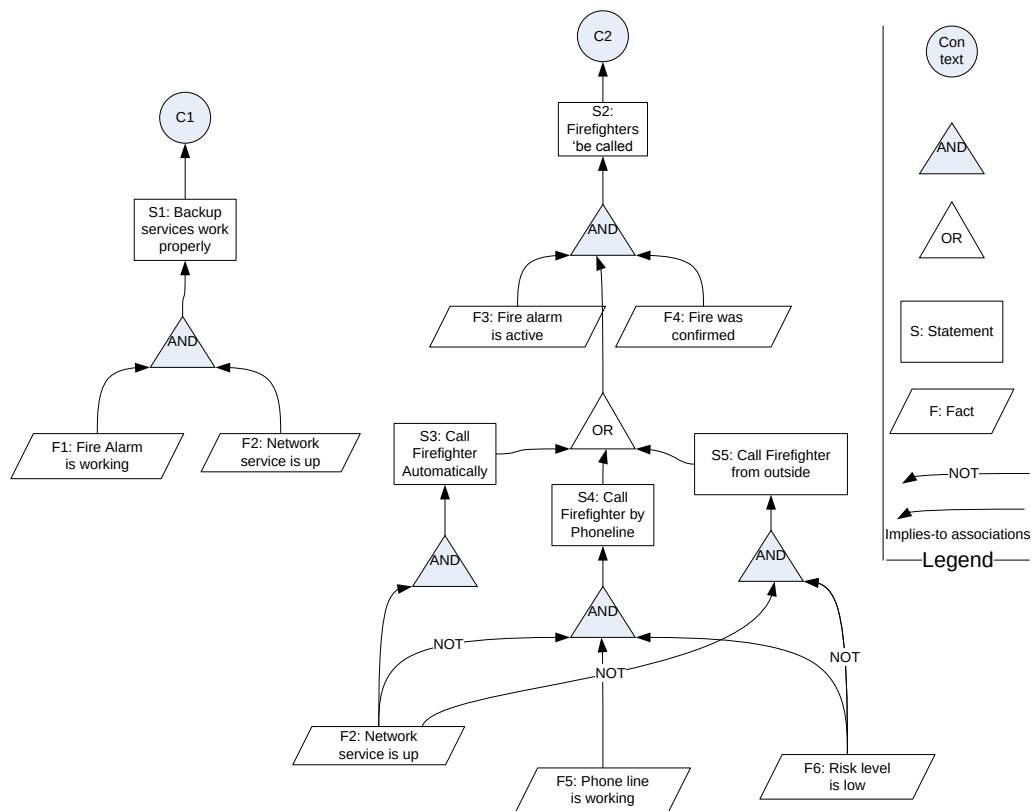


Figure 2.8: Context Information for the Fire Response example

The context representation proposed by Ali et al. (2010) requires the definition of context as formulas that will be evaluated. In Figure 2.8 the context *C1* and *C2* are respectively implied by the statements S1, Backup services work properly, and S2, Firefighters be called. Each statement is supported by a decomposition that ends in facts. The facts such as *Fire Alarm is active* may be evaluated as true or false and the statements supported by this facts may validated.

Spatial modeling approaches Space is an important context in many context-aware applications (Bettini et al. 2010). It is related to locational information what includes describing where the events occur, and what are the resources nearby. In general the spatial context modeling approaches are based in geometric coordinates or symbolic coordinates. Geometric coordinates can be used to specify ranges by their geometric extensions, thus allowing range queries (Bettini et al. 2010). In contrast, symbolic coordinates are represented by identifier, such as a room number or ID. There is no spatial relation offered by symbolic coordinates. In order to allow spatial reasoning about ranges and distances explicit information about the spatial relations need to be provided.

Conceptual (ontology-based) The ontology-based context modeling approaches rely in the symbolic knowledge representation of the contextual information (Nunes et al. 2009). The benefits of using ontologies are the following (Bettini et al. 2010, Nunes et al. 2009): the expressiveness of the language can describe complex context data that cannot be represented by other languages; by providing formal semantics to context data, it becomes possible to share and/or integrate context among different sources; the available reasoning tools can check for consistency of the set of relationships describing a context scenario.

Since ontologies are essentially descriptions of concepts and their relationships, they can be a suitable formalism to represent contextual information. However, ontology-based approaches require more effort during the development of context-aware applications. They also require the design of specific architectures to support processing and manage the ontology. Moreover, several layers of analysis should be applied to design, validate and update the ontology itself, what becomes a drawback when compared with other notations.

2.4.2 Contextual Information in Business Process

The business process modeling area has advanced towards the contextualization. Some initial works have presented the basis for the development of business process management applications that are context-aware. In order to show how contexts impact business processes, this section we describe the classification proposed by Rosemann et al. (2008), which considers contextual information in several layers such as immediate, internal, external and environmental layers.

Immediate Context According to Rosemann et al. (2008) the immediate context of a business process includes elements present in the process itself. They go beyond the constructs that constitute the control flows, and are already well considered in the current business process modeling techniques. These elements are essential to the understanding and execution of business process. The immediate context includes (Rosemann et al. 2008): (input, output) data, participants of process (e.g., organizational unit, position, and person) and IT resources (e.g., application system, database, and logistical resources).

Internal Context The next level of context information covers the information on the internal environment of an organization that impacts the process. The internal context includes the corporate strategy and related process objectives. A change in internal strategies will have impact on business processes (e.g., cost-cutting strategies). Other important enabling or constraining factors for the capability to change are also cited by Rosemann et al. (2008), which include the stakeholders and their risk perceptions, communication and logistical infrastructures, and financial and other resources.

External Context The external contexts are located beyond the control sphere of an organization but still reside in the business network of the organization (Rosemann et al. 2008). The elements of external context are not present in the daily process operations but they still pose impact on the way the organization designs and executes its business processes. For instance, external stakeholders such as suppliers, customers, financial and logistical providers are part of the external context. They can affect the process with their strategies, demands, resources, and failures. Other factors also are part of the external context such as industry-specific practices like supply chain management practices. Since this layer is the boundary between the organization and its partners, the external context will demand compliance of

internal business process and provide a set of constraints that have to be considered.

Environmental Context According to the classification of Rosemann et al. (2008) the outermost layer in the business process context is the environmental context layer. It resides beyond the business network in which the organization operates. The environmental contexts affect the business process with contingency effect. Several factors are part of the environmental context such as weather (e.g., increasing call volume during storms), time (e.g., different sales volume during holidays like Christmas) and workforce related (e.g., worker shortage or strike). Some of these factors can change regularly and impact the business processes (e.g., weather conditions), whereas some may have a long life span (e.g., availability of natural resources, currency).

The business process contexts as classified by Rosemann et al. (2008) shows how deep the contexts can affect the business processes. From data of the process itself to events outside of the organization boundary several levels contexts may require adaptation of the process to a new set.

2.4.3 Context in Business Process Modeling

De La Vara et al. (2010a), describes an approach to include contextualization in business process models. In many ways their work is the most similar to our approach. Since it uses contextualization analysis, as proposed by Ali et al. (2010), their approach incorporates the concepts of context, fact and statements to represent the contextual information. De La Vara et al. (2010b) describe a process to introduce contextualization into business process models through context analysis. Since they describe the variability embedded into the contextual description, the variants and variation point are not explicitly declared. Thus, evaluating the inclusion of new variants as well as updating the variability model are challenging tasks. De La Vara et al. (2010a) is focused in the description of contextual information. Even though they describe the process variability in some level during description of contexts, the variability is not explicitly described. In order to provide more flexibility, in this thesis we separate all configuration information from the model itself. The separation between variability and context analysis helps the analysis process and maintainability of our approach. In BVCCoN we apply the NFRs as driven mechanisms to select a configuration, De La Vara et al. are not concerned in this matter. Moreover, the identification variants and the linking with NFR are performed in different steps of the process analysis. In our approach, we deal with the variants and variation

points since the beginning. Since during this step the analyst just need to focus in to analyze the variability, the separation simplifies this analysis.

Bessai et al. (2008) proposed a context-aware business process approach that evaluates and enhances business process models for a context using workflow patterns. They are focused in the redesign of process models through patterns. During the design the use of contexts help analysts to select a specific pattern. Our approach is similar in a way to their work due the use of patterns. However, we have focused on the use of NFR as mechanism to configure the model. In doing so, the reasoning about the decisions is not kept during configuration. In order to do so, our approach requires qualitative analysis to identify the contributions to NFR. What give us information that may be used to optimize and adapt the process. Bessai et al. (2008) select process re-engineering patterns based on contextual information. They are focused on the evaluation of design time selection of patterns, where a business analyst will select how change the processes how they will be created as new. We are also interested in the evaluation of context and patterns during selection as part of a selection mechanism. However, we have a distinct point of view, the use of context as trigger mechanism for adaptation and adopt patterns to structure the change in process models. The main difference is that we are interested in reduce or eliminated the need for human intervention during the process changes. Moreover, the use of NFRs as driven mechanism for configuration is an unique selling point in our approach. Observe that Bessai et al. do not mention cases where the NFRs can actually be relevant in solving conflicts among variants.

Greenwood & Rimassa (2007) developed a suite to monitor and adapt business process models and integrated software applications. Their approach is a goal-oriented business process management system that supports autonomic features. They have proposed their own language and rely on planning algorithms to support the self-configuration features. In contrast, our proposal has been developed over a standard model (BPMN), which allows its integration with case tools that support this standard.

Hallerbach et al. (2008) also propose a methodology to configure process variants based on context information. They model contextual information using context rules that associated variables with changes in the process model. The process models and variants are represented using the Propov method that provides both the process language and the variant representation in the same notation. A reference process is associated with the change actions and possible variants, when the context rules are active the process is changed into a new process. The main differences between our proposal and Hallerbach et al. (2008) are the representation of variability and the focus on multicriteria decision based on NFRs. While BVCCoN separates the variabil-

ity description and business process modeling in different perspectives, the Provop approach (Hallerbach et al. 2008) mixes them into an extended business process modeling language. This integration trades the maintainability for expressiveness of business process modeling language. On the other hand, we can use a standard language and improve the keep the variability description in separate models. If necessary, we can update the BVCCoN model or change variants without large impact in the business process models. Moreover, our approach focus in use of NFRs as configuration criteria. NFRs allow representation of conflicts among NFRs and interference among preferences of stakeholders. With the use of an algorithm that support NFRs analysis the BVCCoN approach decision making takes in account more elements what creates more perspectives for optimization.

2.5 Comparison to Related Work

In recent years the development of information systems that support business processes has grown and become part of the practices of organizations. One of the aspects that has been reported in the literature is the need of some business processes to adapt to the environment. In order to achieve flexibility several factors must be considered. Indeed, processes should be documented through models. Process models allow an abstraction that facilitates the representation and analysis of practices at work. Then there must be mechanisms to describe how the models can adapt to the environment. Finally, a set of criteria must be available to guide the adaptation to the surrounding environment.

In this section we present a comparison that assess the differences among BVCCoN approach and other proposal that have been described in literature. We designed the comparison based on a set of criteria that evaluates specific issues. Following we present the results.

We are comparing configuration approaches of business process models in dynamic environments described in the literature. The BVCCoN approach covers three main points: the variability representation, the context-awareness, and the preferences representation using NFRs. Since there are several proposals addressing at least one of these points, we selected business process models configuration approaches that are related to two or more of these issues.

We are interested in identifying how approaches described the in literature deal with the configuration of business process models and contrast them with BVCCoN approach. Our approach was designed to help in the configuration of business process models in dynamics environments. These environments

are characterized by interferences of external sources in the business process. Thus, using contexts to describe environment changes, and NFRs to represent preferences we intent to improve the process configuration. In doing so, the comparison will show the difference of using the BVCCoN approach in contrast to approaches that cover similar aspects but were not designed for the same scenario.

2.5.1 Methodology

It represents situations where configuration of process is required. Based on this scenario the participants will evaluate three approaches by reviewing the documentation produced. Three participants agreed on participate of the study.

- Ability to express the process models. In this criteria the approaches are evaluated according to their capacity to represent business process models. The approaches may refer to Business Process Modeling Languages that already exist or create new languages.
- Ability to model variability of process models. Several types of approach may be used to represent variability in business processes (Bentellis & Boufaïda 2008). In this criteria the approach will be evaluated according their capacity to represent variability in business process.
- Presence of mechanisms to support change or configuration of process models. The approaches should also provide mechanisms to perform changes in the processes. This criteria assess the presence and impact of these mechanisms.
- Guidance to support changes in the models. One of the essential point in the configuration is how the approach supports and guides the configuration.
- Level of user interference during configuration. This criteria evaluates the way approaches users will interact, and what the importance of the users during the configuration.

The comparison takes in account works that have been proposed to support business process change/configuration. The following publications are the target of the study:

- M. La Rosa, W. van der Aalst, M. Dumas, A. ter Hofstede, Questionnaire-based variability modeling for system configuration, *Software and Systems Modeling*. 8 (2009) 251-274.

- A. Lapouchnian, Y. Yu, J. Mylopoulos, Requirements-Driven Design and Configuration Management of Business Processes, in: Proceedings of the 5th International Conference on Business Process Management, BPM 2007, 2007, p. 246-261.
- JL De La Vara, R. Here, F. Dalpiaz, J. Sanchez, P. Giorgini, COMPRO: A Methodological Approach for Business Process Contextualisation, in: Proceedings of the 18th International Conference on Cooperative Information Systems, 2010.
- E. Santos, J. Pimentel, D. Dermeval, J. Castro, O. Pastor, Using NFR and Context to Deal with Adaptability in Business Process Models, in: Proceedings of the 2nd International Workshop on Run Time Requirements @ 2011, Trento, Italy, 2011.

The scenario designed was the Airport check-in and boarding processes presented in Chapter 5. In this scenario it is easy to identify circumstances where the process configuration would be useful.

2.5.2 Summary of target works

La Rosa (2009)

La Rosa (2009) proposed an approach to capture system variability based on questionnaire models. They support the configuration with a process and a questionnaire model. Since the choices are interdependent, the questionnaire model consists of a graph which defines the questions to be answered. Questions are answered in sequence, leading to the identification of actions that are performed by the configuration. The dependency chain is dealt with the design to avoid cycles and deadlocks during the execution.

The process models are represented by configurable process models written in the Configurable Yet Another Workflow Language (C-YAWL) (Gottschalk et al. 2008). This language is an extension that has marks to describe the process variation points within the process model. The process models with variability information are aligned with the questionnaire model. The configuration process is performed by the user answering the questions and which leads to new process models.

Lapouchnian et al. (2007)

Lapouchnian et al. (2007) present an approach that aligns business processes with goal models. Goal models are used to describe business goals and user preferences. Through decomposition of the goal models they obtain a goal

structure of the business process. The goal model follows a tree structure where a goal is decomposed in subgoals. This is then annotated with behavioral information. A workflow language is used to describe the behavior.

The description of variability information is added to the structure of goal model. Hence, the process information is mixed with annotations. The annotations that describe variability are based on IF-THEN-ELSE, rules as well as logical operators (e.g., AND, OR). The NFRs are described as softgoals and the contributions to softgoals represent user preferences about the possible choices. The variation points are places where there is an OR decomposition. In order to obtain a workflow process model they describe steps of configuration starting with the analysis of softgoal and selection of variants. After the selection, a transformation generates workflow process models in BPEL.

De La Vara et al. (2010a)

De La Vara et al. (2010a) propose a methodology to add contextualization to business process models. In their approach the purpose is to obtain business process models that are context-aware. Their methodology describe several steps and include contextual annotations that is based on the approach of Ali et al. (2010), which supports contextual analysis. The contexts consists of trees with facts and statements that are evaluated to identify when a context is active.

The contextual information is required to trigger changes in the process flow. In order to include the context, they extended BPMN to have contexts embedded in the sequence flows that link the activities. The variability description is added to the process model, where alternative path in the workflow represents the variants.

2.5.3 Comparison

Process models and variability

The first two criteria analyzed in the comparison are the ability to express process models and to model variability of process models.

The ability to model business process and their variability is an important pre-requisite for the successful configuration approach. A proposal with limitations at this points would offer little help when later it needs to deal with the adaptability in dynamic environments. Having considered how the several approaches represent the process model and variability information we can present in Table 2.1 presents a summary of the several languages used.

It is worth noting that the participants of comparison concluded that all approaches met the two initial criteria, i.e., they are able to express process models and represent variability.

La Rosa (2009) uses a specific modeling language to represent the process models (the C-YAWL). This language is an extension of YAWL that supports gates to represent alternatives. As a result part of the variability description is encoded within the process description. The variability representation is complemented by the design of questionnaires that hide the configuration knowledge and the process information from the user. Since all information are mixed, there is no separation between the variability representation and process modeling.

Lapouchnian et al. (2007) have a different view on the process models when compared to other approaches. The process models tasks are depicted as goals in a goal model. Although at the end of process they propose the derivation of BPEL models from the goal model. Their strategy requires a complex analysis of the goal model. Furthermore, the concepts are quite different since tasks in a workflow model are not in the same abstraction level as the goals described in the goal model. Recall that a goal may require several tasks or even entire routines to be achieved (Yu 1995). Their approach simplifies the variability description due the flexibility of goal models to represent alternatives. However, even simple elements, such as sequence flows, requires annotations in the model what makes the model overloaded. See in Figure 2.9 an example of process model expressed as a goal model with annotations. The goal model has a main goal (i.e., *Supply Customer*) that is decomposed in sub-goals. In order to represent alternatives observe that the *Get Order* goal has a X mark when decomposed into *Get Order Standard* and *Get Order Standard & Web* goals.

De La Vara et al. (2010a) extend BPMN to support context-awareness, that is, tasks and sequence flows are extended to support context information. As a result that the process models are no longer represented in the standard BPMN, but in some kind of BPMN extension. Similar to La Rosa (2009) that rely on C-YAWL, they encoded the variability within the process model elements. For example, alternative choices are represented by BPMN gateways and receive context annotations to allow decision among to selected path.

It is worth noticing that in our BVCCoN approach the process model are represented using standard BPMN language. Since we do not use extensions all of our process models, as well as inputs and outputs of the BVCCoN approach are described in the same language. Hence, BVCCoN model make uses of standard BPMN language, i.e., without change its' concepts. Note that the information required to configure the process is stored in separate

Table 2.1: Process model and Variability

Approach	Process Model	Variability
La Rosa (2009)	C-YAWL/YAWL	Questionnaire and C-YAWL
Lapouchnian et al. (2007)	Goal model/BPEL	Annotations in the goal model
De La Vara et al. (2010a)	Extended BPMN	Annotations in the BPMN model
BVCCoN	BPMN	BVCCoN Model

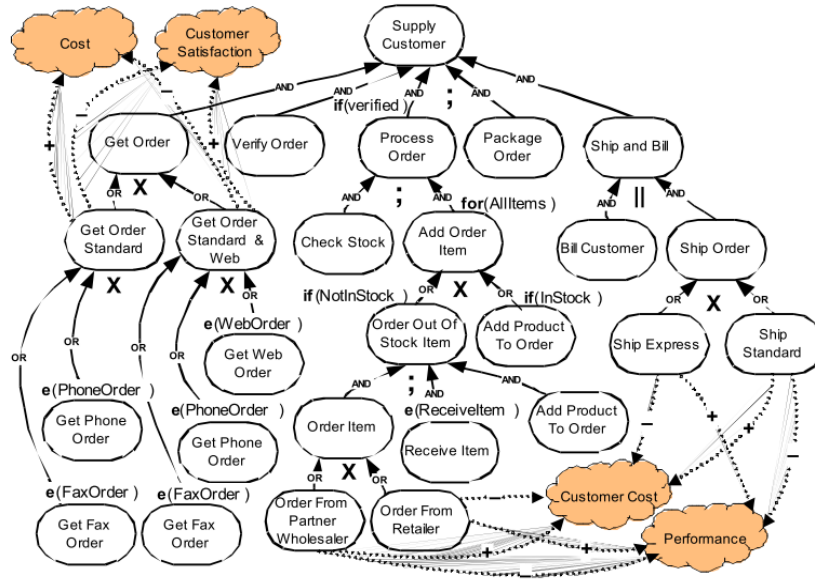


Figure 2.9: Process representation with goal models by Lapouchnian et al. (2007)

models. For example, the variability model covers the basic representation of variability. It links the reference model and relies on process fragments to represent variants.

Considering the way how the process models are represented, the BVCCoN approach, La Rosa (2009), and De La Vara et al. (2010a) require a previous process model to be used as reference model during variability analysis. All these approaches start from a simple process that will be analyzed to obtain and represent variability. Hence, these approaches may require the design of the reference process, which is performed in a separate step. Since Lapouchnian et al. (2007) do not represent the business process using a workflow language, their approach must start with elicitation of a new

process.

De La Vara et al. (2010a) do not propose the derivation of standard and clean models at the end of process. Their approach produces extended BPMN models that holds the annotations at the end of configuration. On the other hand, BVCCoN produces standard BPMN models, as output. The transformation creates BPMN models based on the reference model and includes selected BPMN fragments. Note that La Rosa (2009) require an additional step to remove configuration annotations and derive a process free of annotations. For example, they propose transformation over C-YAWL to derive YAWL. Likewise, Lapouchnian et al. (2007) also require a step to transform from goal model to BPEL language. Thus, both La Rosa (2009) and Lapouchnian et al. (2007) uses two languages to represent the process, i.e., one for the input process and another for output process.

Configuration of process model

The next set of criteria is related to the way the approaches deal with changes, i.e. how they configure process models. Moreover, if the user has clear guidance on how to perform the changes in models. Table 2.2 presents a summary of the approaches according to these two criteria.

In the sequel we analyze the configuration mechanism, how they are used to configure the process models and if some guidance is provided to the user to perform changes in the models. The most basic mechanism is the manual selection, it is simple and common in several scenarios but does not support the user with guidance. The idea to help the user during configuration is to reduce problems with the consistency of solutions and improve performance of the configuration process.

La Rosa (2009) use actions over C-YAWL models to configure the process models. The actions are applied to C-YAWL gates selecting the paths in process that will be part of the final solution. The actions are selected by rules. The input arguments of these rules are the facts present in the answers of the questionnaires. The fact that corresponds to answers are chained to select the sequence of questions that will be answered. Thus, La Rosa (2009) offer a guidance to the configuration by hiding the process information with questions that encode the variants and actions to drive the configuration. The approach prevent the user from selecting the action manually, this improves the intuitiveness of the approach. Figure 2.10 shows an example of the questions and facts used by La Rosa (2009). Note that the questions are associated with facts that are the answer to the question. For instance, the q_3 : *Which Logistics sub-phases have to be implemented?* question is associated with the answers: f_8 : *Freight Tender*, f_9 : *Carrier Appointment*, f_{10} : *Freight*

Table 2.2: Configuration Mechanisms and Guidance

Approach	Configuration mechanism	Guidance
La Rosa (2009)	Nodes selection	Questionnaire chains
Lapouchnian et al. (2007)	Model transformation	Selection quality attributes
De La Vara et al. (2010a)	Context validation	Context validation
BVCCoN	Model transformation	Context validation and selection of quality attributes

the user is not guided to a solution because valid variants may remain in the main process.

In BVCCoN the configurations is performed by selection algorithms and model transformations. During the monitoring stage, the monitored context will validate the presence of variants in the solution. By using NFRs to represent preferences of stakeholders, the selection algorithm chooses variants that will be part of the process. The derivation of the models is driven by a combination of NFRs and context validation. The users describe preferences over alternatives using contribution to NFRs. Moreover, the contextual information allows the adaptation of process. The combination of NFRs and context together with the guidance mechanism free the user from that time consuming task of manually selecting the variants that will be part of derived process.

The results of the comparison for these two criteria is described as follows. Considering the mechanisms to support change or configuration of process models, all approaches cover them in majority of cases. However, when considering the guidance to configuration, some approaches present divergent results. In general, the answers points to the support in majority cases for La Rosa (2009) and Lapouchnian et al. (2007), with one indication each that they could not help the configuration in all possible cases. Observe that a participant states that De La Vara et al. (2010a) would not help in majority of the cases. Compared with other approaches, BVCCoN clearly provides guidance to configuration. The other approaches do not provide a clear guidance to support the configuration of business process.

User intervention

The use of process configuration in dynamic environments implies that immediate responses are required to allow continuous process execution. Thus, the

lesser is the user interference during configuration, the better the approach. Note that the expert that designed the solution may no longer be present and available to take the appropriate action. Table 2.3 shows a summary of the approaches according to this criteria.

La Rosa (2009) proposal is not designed to reduce the intervention of user during configuration. It focuses on the simplification the of configuration by abstracting the complexity of variants' manual selection. During the configuration the user must answer a set of questions to select the variants. Thus, this approach requires an intensive interaction with the user. The user should be an expert in the domain to be able to answer the questions with confidence. This point is a drawback to this approach. In dynamic scenarios, the dependence of human intervention reduces the efficiency in giving an immediate response to changes.

Lapouchnian et al. (2007) proposes the configuration based on the selection of quality attributes. It requires the users to describe their preferences and set the configuration mechanism to obtain a result. The approach may reduce the need for experts in the domain since they are required just to state their preferences over quality attributes. However, their approach was designed to receive these information during configuration, hence it still requires a person to set preferences before the derivation of the process.

De La Vara et al. (2010a) include contextual information in the process, which makes the process independent from user during configuration. Since the configuration is performed by setting the paths that are allowed or blocked, may be the need for the user is not relevant during configuration. Figure 2.11 shows a process with annotations on the flow sequences. These annotations will enable or disable the flow of process. For instance, the *Find potential buyer* sub-process has a gateway where all flows are annotated with contexts formula (e.g., $Fm1 : \neg(F_1 \text{ OR } F_2) \text{ AND } F6$).

The BVCCoN approach supports user intervention to select different priorities for NFRs. However, this information is obtained at design time, i.e. it is not required during configuration. Hence, the user must state their preferences before the monitoring step starts. The contextual change will trigger the configuration based on the priorities of NFRs. Therefore, our approach obtains the configuration for the process without a direct interference of the user during the process.

In all approaches there is a clear difference between the preparation steps and how the configuration is performed. When considering the user intervention all approaches require a domain expert user during the design of the configurable models. In some cases, the same domain expert is also required to perform the configuration as well.

Comparing BVCCoN to La Rosa (2009) it is possible to identify a dif-

Table 2.3: User Intervention

Approach	During configuration
La Rosa (2009)	Required to answer questions
Lapouchnian et al. (2007)	Required to select attributes
De La Vara et al. (2010a)	Not required
BVCCoN	Not required

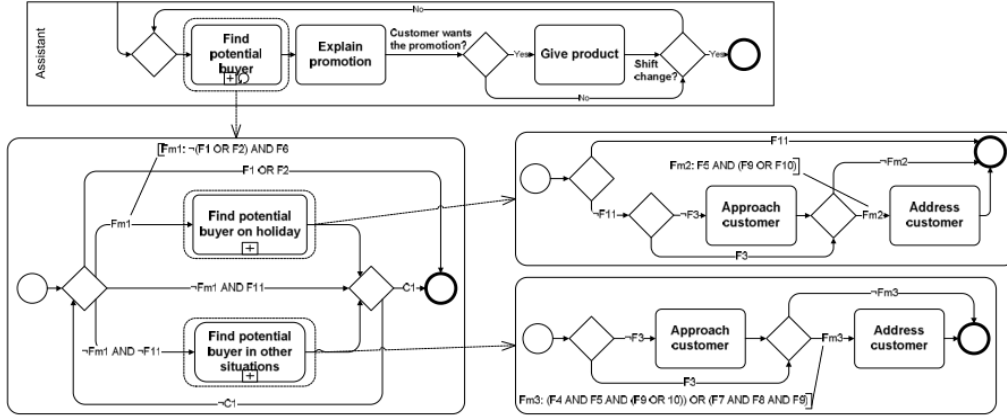


Figure 2.11: Contextualized process models proposed by De La Vara et al. (2010a)

ference in how the user interacts to obtain the configuration. In La Rosa (2009) the domain expert that designed the configurable process do not need to be the same that will configure the model. By adding an abstraction layer with questionnaires, other users may perform the configuration. However, the presence of a skilled user is still required. In contrast, the BVCCoN was designed to perform the configuration using the information stored in the models. Thus, it reduces or even suppress the need for intervention during the configuration step, which is essential for dynamic adaptive environments.

BVCCoN and Lapouchnian et al. (2007) shares the idea that a configuration can benefit from preferences defined over quality attribute. Thus, the user interference is supported by both approaches. The difference lies on the need for an intervention during configuration, Lapouchnian et al. (2007) requires the users to set their preferences each time they want a new process, while BVCCoN does not require user intervention.

We can see that BVCCoN does not require the user intervention to obtain a configured process. Even using a similar mechanism to trigger changes in the model, De La Vara et al. (2010a) do not state how the user should interact during the configuration. Thus, we cannot have a clear comparison between

these two approaches at this point.

According with the results the perception was that the proposals of La Rosa (2009) and De La Vara et al. (2010a) do not reduce user interference. This may be explained by the fact that La Rosa (2009) requires user intervention to perform the configuration as presented in this section. De La Vara et al. (2010a), on the other hand, do not state the need for user to perform configuration. We believe that it is motivated by the lack of details in the document presented as a target of the study. The BVCCoN approach and Lapouchnian et al. (2007) are able to reduce user interference in the majority of cases. The BVCCoN and Lapouchnian et al. (2007) state the user activities. With the user intervention been necessary in some degree by the latter.

2.5.4 Discussion of Comparison

Comparing BVCCoN with the approaches we can see that all are time consuming, which make them not free of effort. The difference in this point is the complexity of their concepts and the way they are used in the business process configuration. Our BVCCoN approach integrates metamodels of the analysis preformed during the BVCCoN process steps, with focus in a clear separation of the concepts. Observe that De La Vara et al. (2010a) uses context variability to define process variability, while BVCCoN does it using a variability model for business process. In BVCCoN, when the context are applied the variation point and variants are already defined. It reduces the context analysis to just relevant contexts avoiding problems with the coverage of the process model and reducing analysis of contexts that will not be used.

When compared with the approach proposed by Lapouchnian et al. (2007) we can see that the integration of metamodels also is an advantage. Observe that in BVCCoN the NFR analysis is performed independently of the variability analysis. During construction of the BVCCoN model the contribution analysis establish links between the variants and NFRs. Hence, the analysis of preferences can be redone if a variant is added or removed. In contrast, the approach proposed by Lapouchnian et al. (2007) mixes the process information with variability description and softgoals. When a variant is added or removed all the model must be reviewed. Different from the other two approaches the La Rosa (2009)'s proposal share less in common with BVCCoN. They use different strategies to deal with variability and configuration. For this reason, while La Rosa (2009) focus in reducing the burden from the user that will perform the configuration, the BVCCoN approach tries to reduce the need for the user interaction during configuration step.

BVCCoN is time consuming but this was a limitation also identified in the other approaches. Some approaches trade the number of analysis steps for logical and mathematical formalism, but we identified by the response of participants, that this is still a serious limitation. Moreover, when dealing with variability even intuitive solutions may become complex as their size grows.

A point that was raised during comparison was the need for tool support. Since some approaches present poor tool support, the benefits of these approaches may be limited. The BVCCoN version used in the study was described in the paper Santos et al. (2011) and is not yet supported by a tool to model. This is a limitation that was partially mitigated with the implementation presented in Chapter 3. However, a complete implementation is not present. The other approaches that do not present tool support also showed limitations. Even so, the comparison was based on document evaluation, thus, the tool support were not evaluated.

In summary, even sharing some common characteristics with the concurrent approaches, the BVCCoN shows more advantages than drawbacks when considering the scenario of dynamic process configuration.

2.6 Summary

In this chapter, we presented the background necessary to understand this thesis. We reviewed Business Process Modeling, Non-Functional Requirements and Contextualization, giving an explanation of the basic concepts and notations of these areas.

The growing integration among business process management and information technology has motivated the study of new technologies for this area. In this way, the Business Process modeling techniques have become important for the development of software applications including the Process-Aware Information Systems (Dumas et al. 2005) and the use of process models as source to obtain software requirements (De La Vara et al. 2010a).

As presented in this Chapters we investigated several business process modeling languages (Aguilar-Savén 2004) and selected the BPMN (OMG 2010a) to represent the business process models. The BPMN (OMG 2010a) is a standard business process modeling language proposed to represent process information in several levels, which includes workflows and choreographies. We select this language due to its wide acceptance by business analysts (Aguilar-Savén 2004). Moreover, BPMN offers resources such as modeling tools and metamodels that are useful for our purposes. Hence, we can reuse tools and analysis methods that are compatible with this modeling language.

According to the literature, approaches that deal with variability for Business Process Modeling are quite advanced and present several mechanisms required representing and manipulating variability. Variability is represented in several ways: Configurable models (La Rosa 2009), Feature Models (Montero et al. 2008*a*), languages extensions (Bayer et al. 2005) and Annotations (Lapouchnian et al. 2007). Other representations of variability found in literature are just variations of these ones. We decide by derive our own solution to represent variability based on the variability description proposed by Pohl et al. (2005). Hence, we improve control over the analysis process and improve modularity of our approach. With that we can obtain a model that consistently fits the analysis process. In next Chapter we detail how we derived a metamodel that is based on this proposal.

When studying the relationship among business process modeling and Non-Functional Requirements, we can see that there are some initial proposals. They cover both the inclusion of NFRs in Business Processes (Pavlovski & Zou 2008) and the use of NFRs as analysis tool to detail the process models (Xavier et al. 2010). NFRs were not well explored as a mechanism to deal with variability, as discussed before, few proposals tried to address this issue (Lapouchnian et al. 2007). However, they were quite initial and still need to be further improved before they are used to online adaptation.

Another important aspect covered in this chapter was the contextualization of business process models. We described the basic definitions of contextualization and how they have been applied to business process modeling. Some approaches have included the context to business process during design with that aim to improve adaptation (De La Vara et al. 2010*a*). Since the contextualization can affects the business processes in several levels, we also described a classification of contexts for business process models proposed by Rosemann et al. (2008). In general these approaches covers the contextualization but still fail to describe quality issues Hallerbach et al. (2008). Although some current approaches can describe quality attributes (in a high level) they still do not consider the interaction among quality attributes and their interference in the representation of stakeholders' preferences.

Chapter 3

Model and Metamodel for Business Process Models Configuration with NFR and contexts

The modeling languages cover several aspects of business process including data, activities and events (Aguilar-Savén 2004). They allow the business analysts to represent the process of an organization, the interaction with external stakeholders, and how to aggregate value with their business. In controlled conditions the business process models describe the exact way the organization should behave. With documentation, understanding and repetition of the processes, the organization tries to achieve effectiveness and productivity (Chang 2006).

When the environment that surrounds the organization changes and impacts the business process, the business analyst must be able to respond to the changes (Bentellis & Boufaïda 2008). Therefore business processes must become flexible. In this thesis we advocate that BPM should incorporate variability information in order to become adaptable to the new contexts and environments. Furthermore, some guidance to support the business process configuration must also be provided.

The goal of this chapter is to provide a variability model which can be applicable to business process models expressed in BPMN language, and integrate Non-Functional Requirements and Contextual information with the variability model, in order to guide the configuration of business process models.

We developed a process called BVCCoN which supports dynamic configuration of business process models. It relies on NFRs and contextual

information. Hence, we also needed to align concepts of different modeling techniques to obtain the configurable models.

In this chapter we present an integrated metamodel that describes the syntax in two levels: abstract and concrete. The abstract syntax provides a representation of concepts and well-formedness rules associated with these concepts, while the concrete syntax describes the diagrammatic representations of the language (Akehurst & Kent 2002). In order to simplify the explanation and avoid the complexity of presenting all metamodel at once, first we present the abstract syntax. After that we present a concrete syntax for a diagrammatic representation of the model, and a concrete syntax using EMF reflexive editor. The concrete syntax will be the basis for a tool support for the modeling technique.

Our approach covers three perspectives in the business process configuration: the variability description, the Non-Functional Requirements, and the context awareness. The first perspective is concerned with describing the variability of business process models and the mechanisms necessary to deal with it. In the second perspective we use NFRs to represent qualities of business processes models; it is concerned with the preferences and interference of quality attributes in the process models. The context-awareness perspective incorporates the influences of the environment in the process models. With the association of monitorable information to business process models, we can provide the capability to adapt them to possible environment changes. These three dimensions are detailed in this chapter. We describe each perspective and their concepts through a metamodel, which provides a structure that allows creating models to support our approach.

3.1 Abstract Syntax

In this section we define the concepts and relationships of the BVCCoN model. The definitions are grouped in four parts: business process workflow model, non-functional requirements model, variability description model, and context model. Some of these perspectives have metamodels defined in the literature such as BPMN and NFR. Our contribution in these cases will be the integration of these metamodels into BVCCoN with the necessary adjustments.

In order to allow the operational use of our approach we need to create models that describe the necessary perspectives. Hence, we designed a metamodel to represent these concepts, their relationships and constraints. The metamodel is described using a metamodeling language that is part of the Eclipse Modeling Framework (Eclipse.org 2012). This language complies

with the OMG's standard MetaObject Facility (MOF). An additional benefit of using metamodeling technology to formalize our approach is the access to the BPMN metamodel. Ecore metamodeling language may be presented in graphical (e.g., class diagrams) or textual (e.g., OCLInEcore) notations. In this chapter we present both graphical and textual representations of Ecore metamodel.

3.1.1 BPMN model

As previous presented in Chapter 2, the BPMN (OMG 2010a) is a standard business process modeling language proposed to represent process information in several levels, which includes workflows and choreographies. We select this language due to its wide acceptance by business analysts (Aguilar-Savén 2004). Moreover, BPMN offers resources such as modeling tools and metamodels that are useful for our purposes.

BPMN modeling tools based on the Eclipse platform use a common metamodel that describes the concepts of BPMN. The metamodel is accessible to any developer. Thus, we use an already validated metamodel used for tool support. Hereafter, when we refer to BPMN metamodel we mean the Eclipse BPMN 2.0 metamodel designed by the Eclipse group (Eclipse.org 2011). Figure 3.1 shows an excerpt of its abstract syntax.

Business process models could involve several types of models, such as workflow models, choreography models and so on. Since our approach is focused on workflow models, when we refer to business process models what is meant is the business process workflow model. In this thesis we assume that the process models will be well defined according with the syntax described by the BPMN specification. The following definition of business process model was borrowed from La Rosa (2009):

Definition 1. *A business process workflow model is a description of workflow elements of a business. The business process models are described in a language that relate elements through relationships. More formally, a business process workflow model describes a tuple $BP = \langle W, T, A, D \rangle$, where W is a set of workflow elements including activities, events and decisions. D is a set of data elements. T is a set of transitions that link workflow elements W or data elements D . A is a set of actors own elements in T , and produce or consume data elements in D .*

Business process models are expressed with the flow of activities, events, decisions, and data that compose the business. These information are internal to an organization. The elements present in this metamodel cover

the basic workflow diagrams, but the BPMN has also elements for other diagrams such as the collaboration and choreography that are not described here. Figure 3.1 shows the hierarchical decomposition of *FlowElement* in BPMN 2.0 metamodel. Due to the size the BPMN metamodel, we suppressed the associations to simplify the presentation. The main element is the *FlowElement*, which is the parent element for *Event*, *Activity*, and *Gateway*. When the other perspectives need reference elements of BPMN they point to the *FlowElement* and use OCL rules to evaluate their sub-types. An *Activity* may be specialized by two types an atomic activity represented by a *Task* and a composed activity represented by the *SubProcess*. Moreover, a *Task* may be specialized in subtype such as *SendTask*, *ReceiveTask*, *ManualTask* and *ServiceTask*. Data is represented by *DataObject* and has its own link (not present in this excerpt). *FlowNode* are linked by *SequenceFlow* which represent transitions. The *Event* may also start (*StartEvent*), interrupt (*IntermediaryThrowEvent* and *IntermediaryCatchEvent*) and end (*EndEvent*) a workflow. The purpose of Figure 3.1 is to show that all elements used in our approach are derived from the main element in this Figure. Thus, hereafter in this chapter the metamodels will refer to the *FlowElement* instead to specific BPMN elements.

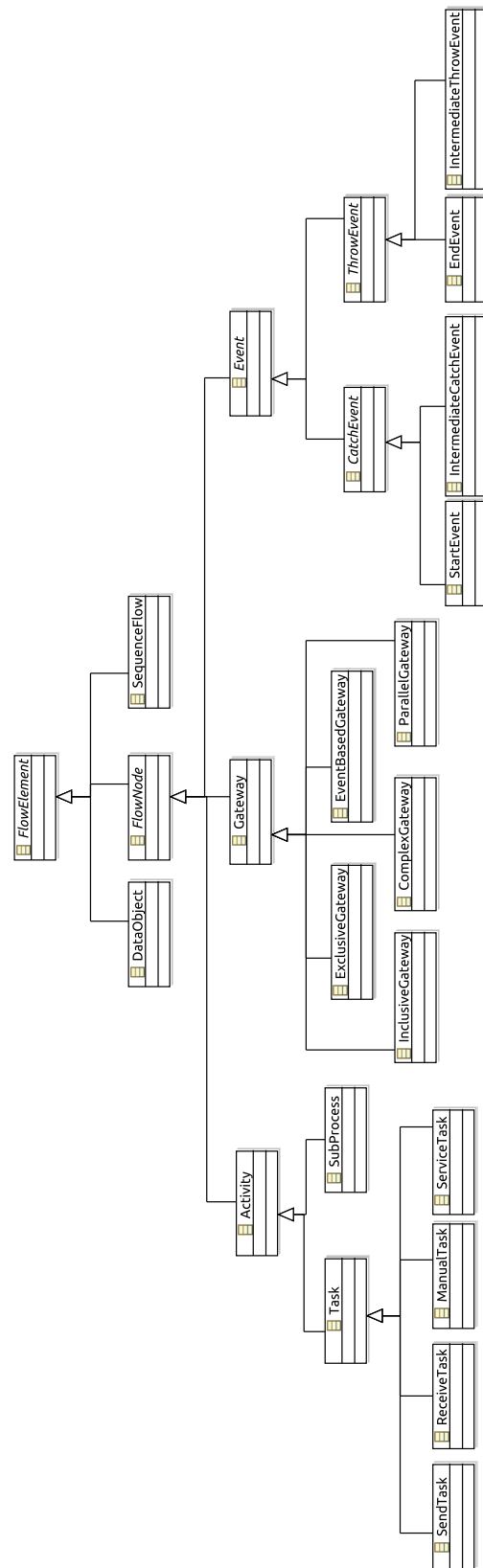


Figure 3.1: BPMN 2.0 metamodel excerpt adapted from Eclipse.org (2011)

3.1.2 Variability

The variability modeling in business processes models represents alternative ways to perform its activities, who will be responsible for performing them, which resources are required by and consumed by (La Rosa 2009). We rely on variants and variation points to describe the desired variability. Variation points are the subjects of change, while variants are the objects of change (Pohl et al. 2005). Since we are concerned with explicitly representation of variability we adopted the variant and variation points concepts. They are well defined in SPL literature and may be translated to several artifacts (Pohl et al. 2005)(e.g., documents, models, program code). Hence, we believe they are suitable to represent business process variability as well. In the case of business process models, the variability can be expressed according with the way the process is described. Since we are dealing with business process workflow models, the variability represents the alternative or variable ways that the activities in the workflow may be performing. It may also include the variability in events and data representation that are present in the business process model.

Before defining our own view of variability we revisit the well-known variability description of Pohl et al. (2005) and evaluate its concepts. The variability concepts proposed by them are described in the metamodel of Figure 3.2. According with this metamodel, the variability model must cover variants, variation points and relationships such as the constraint dependency among variants, constraint dependency among variation points, and constraint dependency between variants and variation points. Those authors employ the terms *Requires* and *Excludes* in all the constraints relationships. The *Requires* relationship assumes that in order for an element of variability to be part of the solution it will also need the presence of the required element, but the inverse does not hold. On the other hand, when the *Excludes* relationship is used, both variability elements are mutually exclusive, that is, if one is selected the other cannot also be present.

Aside from the constraints relationships there are also some other variability dependency relationships: *Mandatory*, *Optional*, and *Alternative Choice*. The *Mandatory* relationship means that all of variants should be part of the solution, while the variants in the *Optional* relationship may be part (or not) of the solution. On the other hand the *Alternative Choice* requires that one and just one of the variant to be part of the solution.

Below, we describe how our metamodel covers these concepts and constraints. The two main concepts that must be represented in the variability perspective of the metamodel are the *VariationPoint* and *Variant*. The former indicates the point of the business process model that can change, and

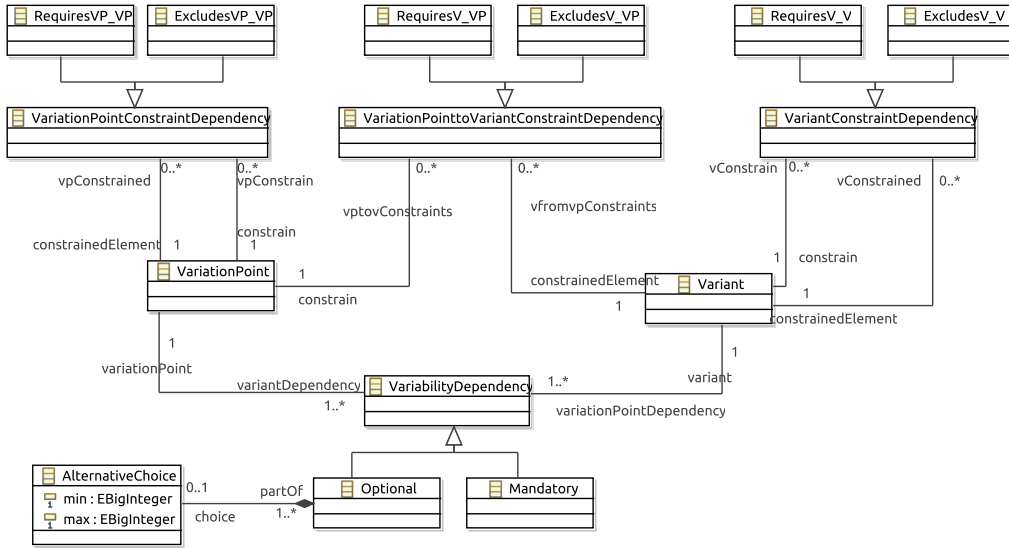


Figure 3.2: Variability Metamodel adapted from Pohl et al. (2005)

the latter are parts of process that can take place in the *VariationPoint*. Because in this thesis we are dealing with process models, then both *Variants* and *VariationPoints* are represented in terms of business process elements (see Figure 3.3).

The description of a *VariationPoint* includes an unique identifier (the id attribute) and a name. They are used to refer to *VariationPoints* during the modeling process. The first feature that relates our variability metamodel with the BPMN metamodel is the reference points *begins* and *ends*. A *VariationPoint* indicates the place of the BPMN model where the *Variants* may be placed. Both are of the type *FlowElement*, already present in the BPMN metamodel.

In the same way as the *VariationPoints*, the *Variants* also have an unique identifier and name. The *Variants* can be associated to one or more *VariationPoints*. To represent the variability dependency relationships we use the *operator* attribute. The relationships *Optional*, *Mandatory* and *Alternative Choices* are represented, respectively, by the operators OR, AND, and XOR. Associated with these operators we employ the use of patterns. workflow patterns described in the literature (see Section 2.1.5) are useful to describe how the process variants can be inserted in the variation points. The *WorkflowPatterns* may represent sequence, parallelism, optional behavior, and so on. Indeed, some authors argue that the use of patterns may help to preserve the correctness of a process (Wohed et al. 2005).

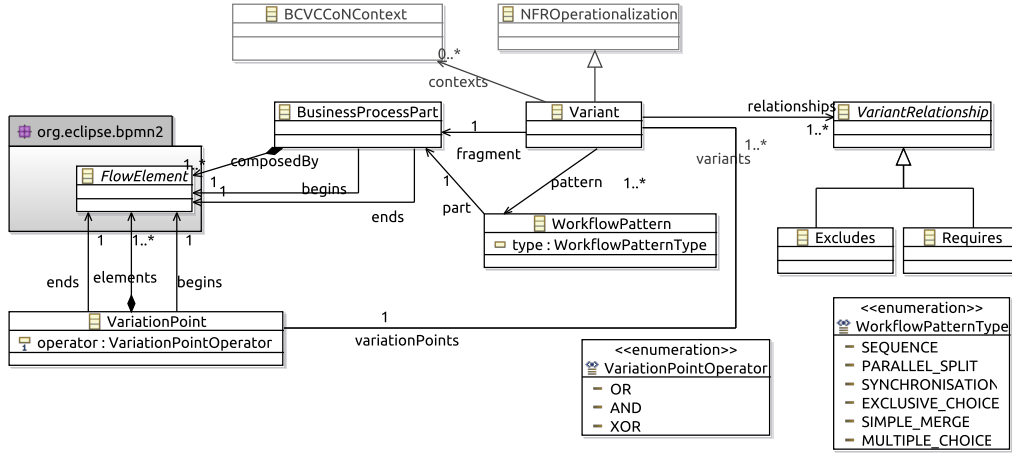


Figure 3.3: Variability perspective of BVCCoN

The *Variant* is linked to *VariationPoints* through *Patterns*. The *Variants* are represented by fragments of process model in the BPMN language, the *BusinessProcessPart*. The analyst that is responsible for the variability description must describe the consistent fragments of process model that can be placed in a *VariationPoint*. The idea to use fragments of process model instead of using atomic elements such as activities or events was to improve the flexibility of our representation. First, the process models represent control and information flows that are more difficult to represent using tree-like data structures common in representation of variability (e.g., Feature Models). Hence, we avoid complex variability controls such as the use of annotations to represent sequences and decisions as presented by Lapouchnian et al. (2007). The consistency check is performed by the analyst that at design time will have the opportunity to identify possible semantic problems among variants. Note that the granularity of the process in the *BusinessProcessPart* is controlled by the analyst in charge of describing the business process. Each *BusinessProcessPart* refers to *FlowElements* of BPMN metamodel and must match with the *begins* and *ends* points present in the description of the *VariationPoint*.

Another aspect of the *Variant* that is important to be represented is the relationship among *Variants* in different *VariationPoints*. According to the variability metamodel of Figure 3.2, a *Variant* can have two types of constraints one that requires or excludes *Variants* and another that requires or excludes *VariationPoints*. The former is partially addressed when a *Variant* of a *VariationPoint* requires the presence of a *Variant* of other *VariationPoint*. This will force the existence of *VariationPoints* that are required

by *Variants*. However, the *Excludes* behavior is not addressed in our solution. The latter is achieved through the *VariantRelationships Requires* and *Excludes*.

3.1.3 Context-Awareness

Context is the current state of the user or system to the objects in their surrounding environment (Bai et al. 2008). In order to define the abstract syntax of BVCCoN contextualization perspective, we evaluated the proposal of Ali et al. (2010), and translated it into a metamodel which was integrated with our approach. We adopted Ali et al. (2010) approach due to its support for the logical evaluation of context expressions. By using this approach as reference we benefit from the reasoning mechanisms proposed by Ali et al. (2010). These mechanisms allow verifying the consistency of variability model according the validity of contexts, what could help the business analysts during design of BVCCoN models. Moreover, their proposal has been applied in business process modeling (De La Vara et al. 2010a), thus, it is suitable for our scenario.

The context representation language proposed by Ali et al. (2010) is a fact-based language. As discussed in Section 2.4, it is based in the decomposition of context into a formula of facts and statements. A statement is a predicate of world that cannot be verified by an actor in the context. The statement may be decomposed in other statements or in facts. On the other hand, facts can be verified and associated to variables. Hence, after completing the context analysis we obtain logical expressions that may be assessed as valid.

Based on the elements of the information from Ali et al. (2010) we defined the metamodel of Figure 3.4. Our main contribution in this section was the derivation of a metamodel from the formula presented in Ali et al. (2010). In doing so we could integrate their concept of context to the variability model presented in Figure 3.3. The first concept is *BVCCoNContext* that represents the contextual information of BVCCoN. The context is implied by a *ContextExpression* that is composed by a formula of *Predicates*. A *Predicate* may be a *Statement*, a *Fact*, or a *PredicateDecomposition*. A *Statement* describes an expression that cannot be validated alone. Hence, it is supported by another *Predicate*. When necessary a *Predicate* may be decomposed using AND, OR and NOT operators. Since a NOT decomposition is unary, we constrained it using an OCL expression as presented in Figure 3.5 at lines 4-5.

According with the proposal of Ali et al. (2010) a *Fact* may be assessed and validated. Thus, the values evaluated by the formula of predicates are

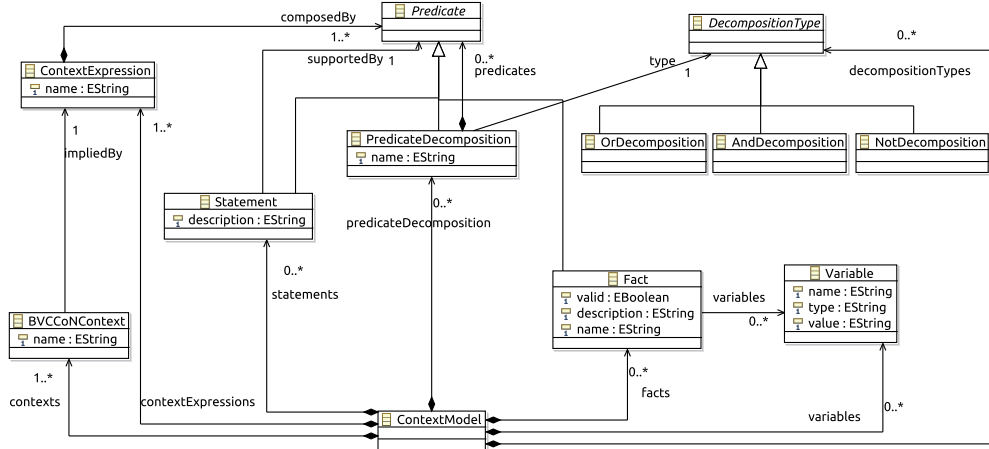


Figure 3.4: Context decomposition metamodel based on Ali et al. (2010)

```

1 class PredicateDecomposition extends Predicate {
2     property predicates : Predicate[*] { composes };
3     attribute type : DecompositionType;
4     invariant NotDecomposition:
5         type = DecompositionType::NOT implies predicates->size ()
6         = 1;
7 }

```

Figure 3.5: PredicateDecomposition with OCL

expressed by a *Variable* that is linked with a *Fact*. When an expression is composed, they may be evaluated to identify the validity of the context.

The context representation proposed by Ali et al. (2010) requires the definition of context as formulas that will be evaluated. In Figure 2.8 the *C1* and *C2* contexts are respectively implied by the statements: S1, Backup services work properly, and S2, Firefighters be called. Each statement is supported by a decomposition that ends in facts. The facts such as *Fire Alarm is active* may be evaluated as true or false and the statements supported by this facts may validated.

3.1.4 Non-Functional Requirements

Another perspective that is covered by our approach is the use of Non-Functional Requirement (NFR) as a way to represent preferences. In this thesis, we adopted the NFR Framework (Chung et al. 2000) to model catalogs and the relationships among NFRs. The NFR Framework is a well know

approach to model and analyze NFRs. We adopted this framework due to its features to model and analyze NFRs. In fact, the idea to represent different levels of satisfaction using contributions to NFRs matches our conception of preference representation. In this section, we will present the basic concepts of NFR Framework and show how we adapted their approach to BVCCoN.

In order to integrate NFR Framework we first designed a metamodel in Ecore language based on the OME 3 Telos metamodel (Mylopoulos et al. 1992). However, the NFR metamodel in Telos did not separate the description of abstract and concrete syntax. Thus, our translation became quite simple due to the elimination of diagrammatic information related with the tool support (i.e., OME 3). The result is presented in Figure 3.6.

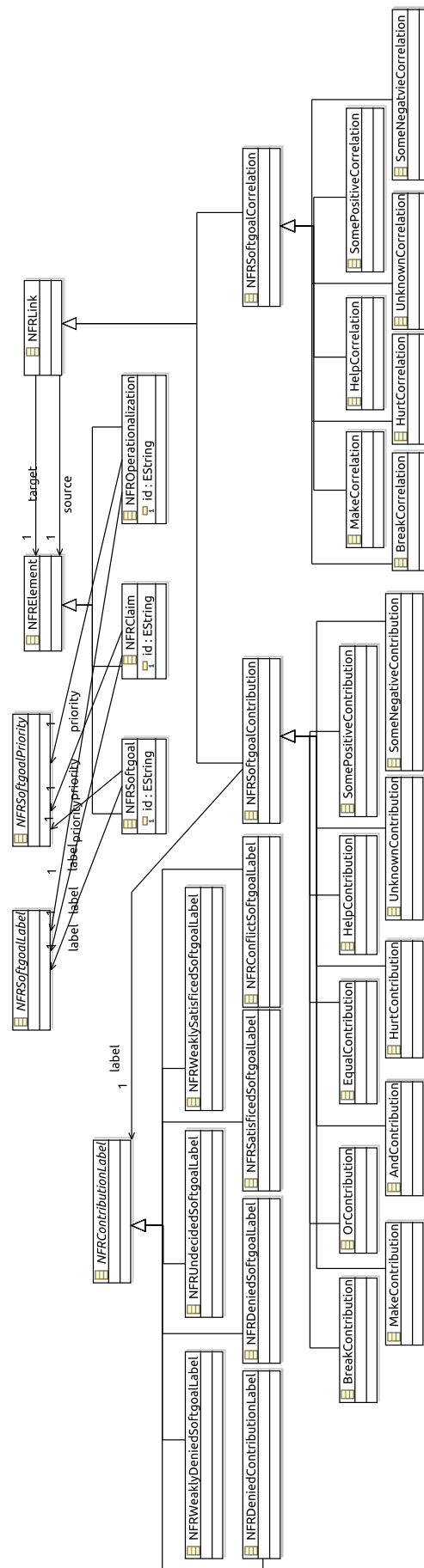


Figure 3.6: NFR Framework metamodel

The main concepts are *NFRElement* and *NFRLink* that represents respectively the elements and their relationships. The *NFRElement* is specialized in *NFRSoftgoal*, *NFRClaim*, and *NFROperationalization*. The relationships are *NFRSoftgoalContribution* and *NFRSoftgoalCorrelation*. Note that they also are specialized in several other types of links as will be explained ahead. In order to support the analysis process some other concepts were defined such as: the *NFRContributionLabel* that indicates the degree of satisfiability of a NFR; the *NFRSoftgoalPriority* that helps with the prioritization during analysis; and *NFRSoftgoalLabel* that serve as description of the NFR.

After the analysis of the NFR Framework metamodel, we identified the elements that are important for our approach and the ones that are not applicable. Thus, during the integration with the BVCCoN metamodel we removed some concepts and adapted other while trying to preserve the main structural concepts. The main concepts are the *NFRSoftgoal* and *NFRSoftgoalContribution*, they compose the main structure of the graph used in NFR Framework. In BVCCoN model, we used them to describe the NFR catalogs, which structure the NFR decompositions, and to represent the contribution from *Variants* to NFRs. Figure 3.7 shows the BVCCoN NFR perspective with a simplified version of NFR and their relationship with other BVCCoN concepts.

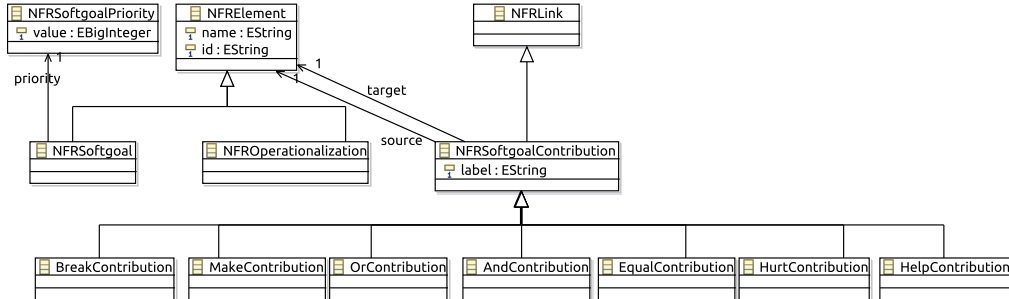


Figure 3.7: NFR perspective of BVCCoN

The representation of preferences involves two elements. On one hand the NFRs that are represented by *NFRSoftgoals* and on the other hand of this relationship are the *Variants*. The NFR Framework has a very similar idea to link a NFR to a functional/concrete element called *NFROperationalization*. What the *Variant* does is to take advantage of this idea, but instead of assuming the operationalization of the NFR we use the contributions to provide the preference's notion.

The contribution relationship represented by *NFRSoftgoalContribution* has several subtypes. Some represent a range of intensity such as *Break-*

Contribution, *HurtContribution*, *EqualContribution*, *HelpContribution*, and *MakeContribution*, which vary respectively from the most negative impact to the most positive impact. Other contribution were designed to provide structural capability to the NFR graph such as *OrContribution*, and *AndContribution*. Likewise, there are also the one provided to represent uncertainty such as *UnknownContribution*, *SomePositiveContribution*, and *SomeNegativeContribution*. These last ones were not included in BVCCoN metamodel because, so far we are not doing analysis that needs them. The others were incorporated.

The *AndContribution* and *OrContribution* were used to decompose the NFRs during the definition of catalogs. In order to avoid problems with elements related to meaning of the contribution they were constrained using OCL rules to be used just among *NFRSoftgoals*. The other contributions (e.g., *MakeContribution*) were constrained to just be applied between a *NFRSoftgoal* as target and a *Variant* as source. Figure 3.8 present the constraints over *NFRSoftgoalContribution*. Observe that the only target allowed is the *NFRSoftgoal* (lines 2-3), the sources of contribution can be *NFRSoftgoal* when it is *AndDecomposition* and *OrDecomposition* (lines 4-6), or a *Variant* in other cases (line 7).

```

1  class NFRSoftgoalContribution {
2      invariant targetNFR:
3          target.ocIsKindOf(NFRSoftgoal);
4      invariant andOrDecomposition:
5          if self.ocIsKindOf(AndContribution) or self.
             ocIsKindOf(OrContribution) then
6              source.ocIsKindOf(NFRSoftgoal)
7          else source.ocIsKindOf(Variant)
8          endif;
9      invariant notTheSameTargetAndSource: source <> target;
10 }
```

Figure 3.8: NFRSoftgoalContribution with OCL

During the configuration process it will be necessary to select the priority of a NFR to perform the selection of variants. Since the NFR Framework already provides this feature, through the *NFRSoftgoalPriority*, we just made some changes in the way it is used. In Chung et al. (2000) the priority is designed to help the analyst to select what is the next branch of a NFR graph to be analyzed. In contrast the BVCCoN presents the priority as a transient attribute of NFRs that may change during execution.

Note that some analysis that are supported by the NFR Framework were not necessary in the BVCCoN approach. To keep a coherent set of con-

cepts in BVCCoN metamodel we pruned the NFR Framework metamodel during the integration and removed these concepts. Even though we work with a very close idea, the BVCCoN does not make analysis of satisfiability, so concepts such as *NFRContributionLabel* and its subtypes, *NFRClaim*, and *NFRCorrelation* and its subtypes were not included in the BVCCoN metamodel.

3.2 Concrete Syntax

In the previous section we described an abstract syntax for our approach models. In this section we will describe the concrete syntax for the variability, NFR and context perspectives. The business process model already has a concrete syntax described in the BPMN 2.0 metamodel, so it is not necessary for us to describe the BPMN concrete syntax.

In BVCCoN model, the concrete syntax involves two proposals of syntax a standard diagrammatic one and alternative structure to be used in an EMF reflexive editor. The diagrammatic concrete syntax is a proposal of a graphical representation of BVCCoN models, how they should look, and their relationship with the metamodel. Moreover, we present the EMF reflexive editor syntax that shows tree-like structure that is equivalent to the model itself.

In order to define the diagrammatic elements of the concrete syntax we employed the OMG Diagram Interchange facility (OMG 2006), described in Ecore. The definition of these elements is default to UML description, and helps with the definition of basic graphical elements. We associated an extension to every element of our abstract syntax with a root element called *BaseElement*. Hence, the elements of diagrams such as shapes, links and so on reference the elements of the abstract syntax metamodel by means of the *BaseElement*. Figure 3.9 presents the elements of Diagram Interchange for BVCCoN and their association with the other parts of metamodel. Note that all graphical elements are associated with the *BaseElement* from the metamodel and all basic elements such as *Variant*, *VariationPoint*, *BVCCoNContext*, *NFRLink*, and *NFRElement*, inherit from *BaseElement*. The elements of diagram such as *BCCoNDiagram*, *BVCCoNPlane*, *BVCCoNShape*, *BVCCoNEdge*, and *BVCCoNLabel*, inherit from the Diagram Interchange metamodel correspondent elements such as *Diagram*, *Plane*, *Shape*, *Edge* and *Label*. This strategy simplifies the definition of the concrete syntax because we just need to describe how the visual element will look (e.g., diagram, link or shape).

Following, we describe the look and feel of the visual elements for BVC-



3.2.1 Variability perspective

The *VariationPoint* is associated with BPMN flow elements. In the abstract syntax the *VariationPoint* refers to elements in a business process model to identify the elements that are part of a *VariationPoint* and the elements that indicate where the *VariationPoint* begins and ends. In the concrete syntax we defined that the *VariationPoint* will have a graphical representation that will store the information associated to a *VariationPoint* such as name, and operator (i.e., AND, OR, XOR). The mark to describe BPMN elements is expressed as a container that will encapsulate these ele-

ments. In order to identify the start and finish of a *VariationPoint* one or more elements of this set will be connected by links with the label *begins* and *ends*.

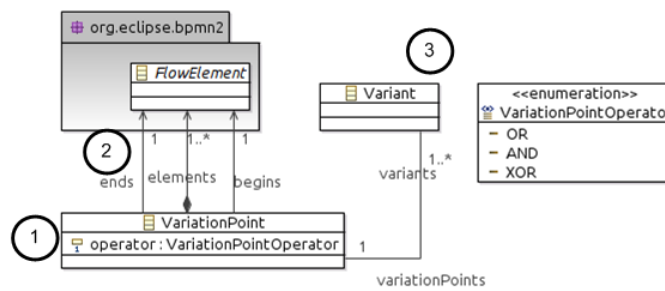
The association between a *VariationPoint* and a *Variant* is expressed by a link. Hence, the source of a link is a *Variant* and the end is a *VariationPoint*, but it is also navigable in the opposite direction.

In order to express how the *Variant* will be placed in the new process it is necessary to describe the *WorkflowPattern*. Instead of a link the *WorkflowPattern* is represented by a property of *Variant*. The most critical *WorkflowPattern* is the relationship among *Sequence* and *Parallel* workflow pattern, both have the meaning of AND. However, the way they are composed is different, when the *WorkflowPattern* parallel is present in all *Variants* of a *VariationPoint* it will be placed using this pattern. We hide the graphical representation of a *WorkflowPattern* that will be part of the properties of *Variant* in both concrete syntax. Thus, during the association of a *Variant* and the *BusinessProcessPart* the *WorkflowPattern* must be set.

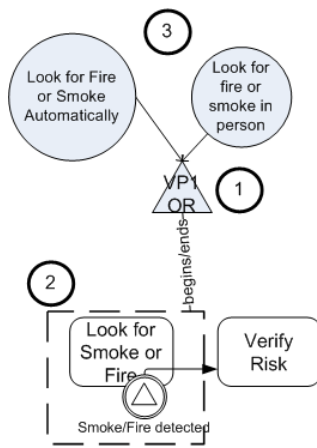
Another characteristic of the *Variant* is the link to business process elements. A *Variant* is associated to a *BusinessProcessPart*. Each *BusinessProcessPart* is modeled by a BPMN modeling tool, the association of their elements is done by links that indicate the elements of where the *Variant* begins and ends, and a container that indicates the intermediary elements that are part of variant. The *Variants* may interact with each other through the links of requires and excludes labels that corresponds to *Requires* and *Excludes* concepts.

The alternative concrete syntax represented in the EMF reflexive editor is generated by model transformation from the Ecore metamodel. It looks like a tree with the elements decomposed in each level of the tree. Multiple association such as the *Variants* present in a *VariationPoint* will look like a list.

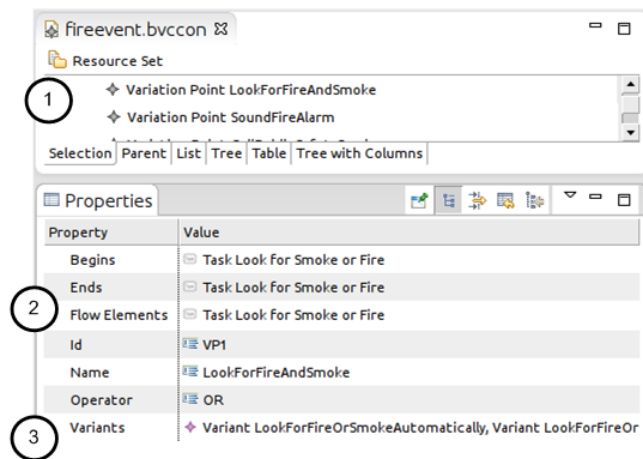
Figure 3.10 shows the relationship among elements of the abstract syntax (a), the graphical concrete syntax (b) and the EMF reflexive editor (c) for a *VariationPoint*. According to our running example of a fire event (see Section 2.3.3), observe that the *VariationPoint* (1) is connected to *FlowElements* (2) represented by *Look for Fire or Smoke Task*, and is associated with *Variants* (3) such as *Look for fire or smoke automatically* and *Look for fire or smoke in person*.



(a) Abstract Syntax



(b) Graphical Concrete Syntax

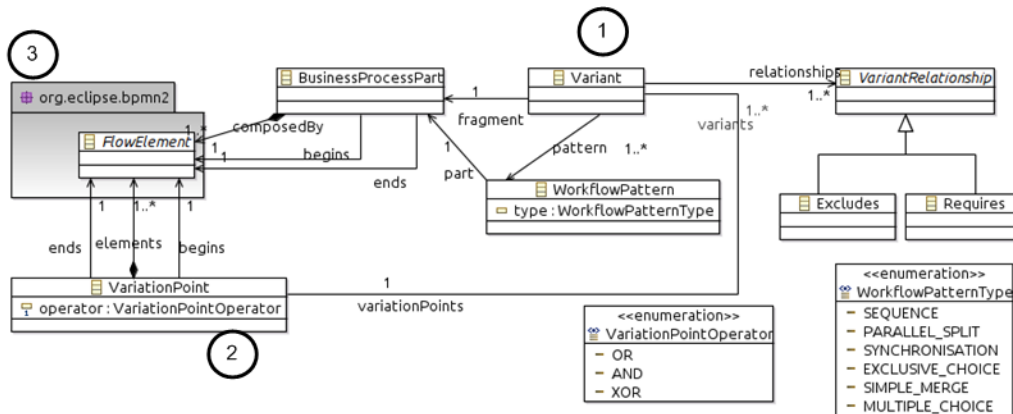


(c) EMF Reflexive editor Concrete Syntax

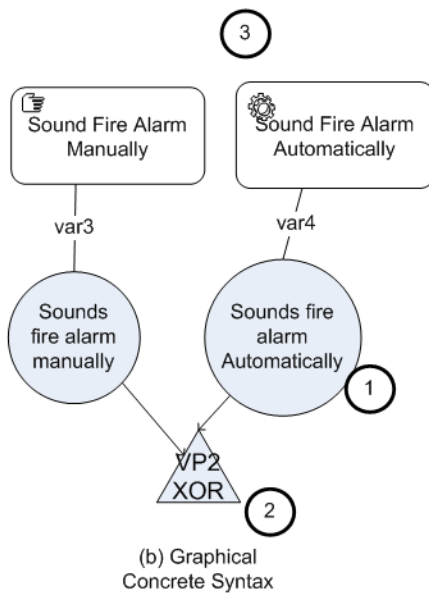
Figure 3.10: Concrete Syntax for VariationPoint

The *Variant* concept shows a complex set of relationships such as the link with business process elements, contexts, and patterns. Figure 3.11 shows two *Variants* associated to a *VariationPoint*. Observe that the *Sound Fire Alarm Automatically Variant* (1) is connected to the *XOR VariationPoint* (2), and is associated to a *BusinessProcessPart* (3) that is represented by the *Sound Fire Alarm Automatically Task*.

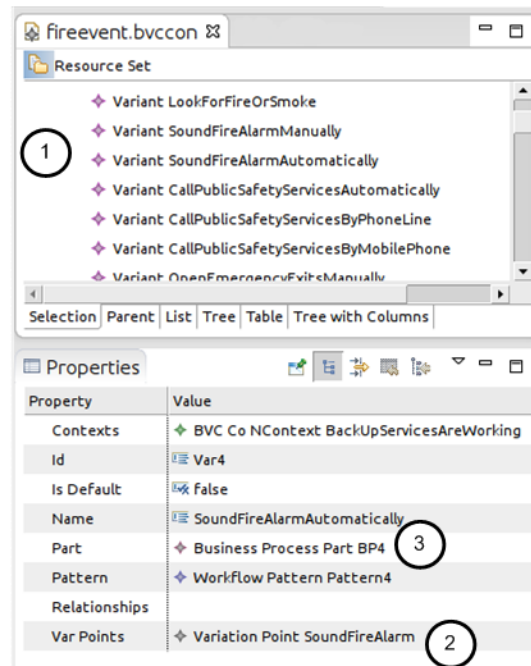
It is important to highlight that the use of triangle and box shapes do not have relationship with other parts of modeling language. The BPMN modeling elements are dealt in different metamodel, as well as the context decomposition model may present similar shapes but with different meaning. During the modeling both context decomposition and BPMN are modeled in different modeling environment, thus, even with similar shapes no intersection will happen during variability modeling.



(a) Abstract Syntax



(b) Graphical Concrete Syntax



(c) EMF Reflexive editor Concrete Syntax

Figure 3.11: Concrete Syntax for Variant

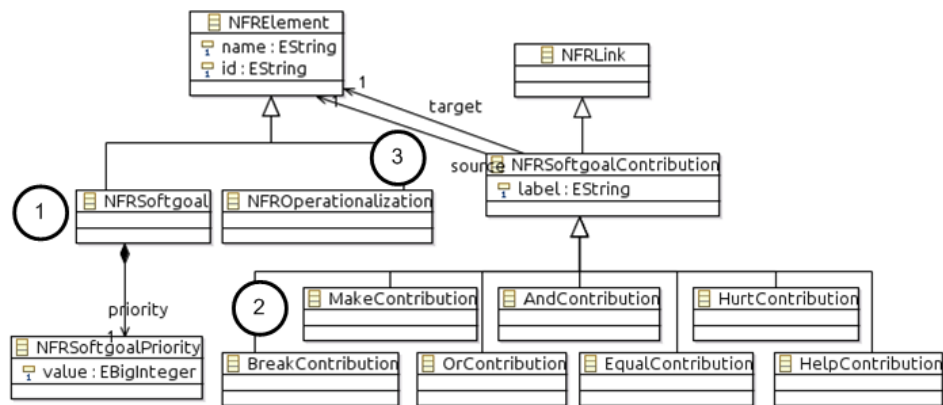
3.2.2 NFR perspective

In the NFR perspective, we address the problem of adapting the concrete syntax of BVCCoN to the representation provided by Chung et al. (2000). In the NFR Framework the concrete syntax is described in a Softgoal Integrated Graph (SIG). A SIG is a diagram where the *NFRSoftgoals* are nodes of graph, represented as clouds, that are associated by the *NFRSoftgoalContribution* and *NFRSoftgoalCorrelation* links. We maintain the visual representation of a cloud in the graphical representation of a *NFRSoftgoal*.

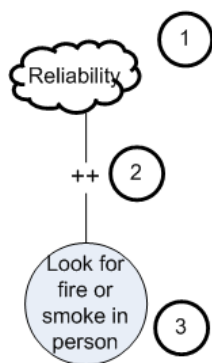
When *NFRSoftgoalContribution* connects two *NFRSoftgoals* we use the *AndContribution* and *OrContribution*. Both are represented by links with AND and OR label respectively. On the other hand, when they are associating *NFROperationalizations* and *NFRSoftgoals* the contributions are *MakeContribution*, *HelpContribution*, *HurtContribution*, *BreakContribution*, and *EqualContribution*. We also propose the graphical representation of a link with the ++, +, -, --, and = labels instead of the correspondent textual labels. This change was required to allow a simpler association by the analyst with the idea of preference, since labels such as *Make* or *Break*, could cause misconception in this point.

The representation of a *NFROperationalization* is the same as the *Variant* representation, since the *Variant* class inherits from *NFROperationalization* class. The SIG representation of a *NFROperationalization* is a cloud just as the of *NFRSoftgoal* with a stronger board line. By using the *Variant* representation we help to distinguish among a *NFRSoftgoal* and a *NFROperationalization*.

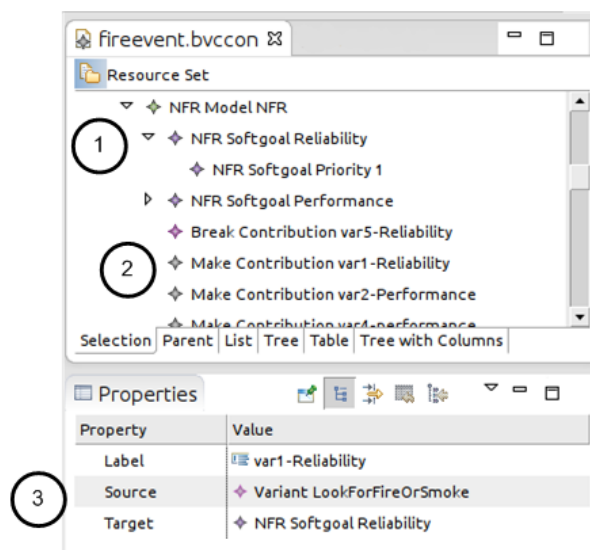
Figure 3.12 describes the elements of the abstract syntax (a), the graphical concrete syntax (b), and the EMF reflexive editor (c). A *NFRSoftgoal* (1) is represented by a cloud with the *Reliability* label, it is receiving a *MakeContribution* (2) from the *Look for Fire or Smoke Variant* (3). Observe that the representation of a contribution in the EMF reflexive editor (c) has the properties source and target (3) that correspond to where the link anchored.



(a) Abstract Syntax



(b) Graphical Concrete Syntax



(c) EMF Reflexive editor Concrete Syntax

Figure 3.12: Concrete Syntax for NFR

3.2.3 Context perspective

The context description in the concrete syntax is performed in two representations, the analysis of contexts and the association with *Variables*. We begin describing the analysis of a *ContextModel*. A *ContextModel* is formed by *BVCCoNContexts* that are the root nodes for the analysis. Each *BVCCoNContext* is implied by a *ContextExpression*. In the graphical representation the *BVCCoNContext* and *ContextExpression* are presented in the same visual, but in the EMF reflexive editor their representation are separated.

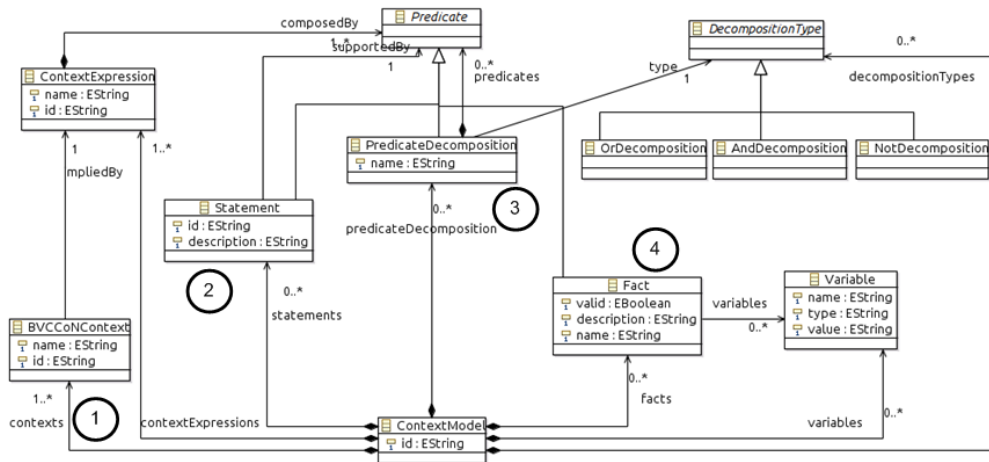
A *ContextExpression* is composed by a formula of predicates. A *Predicate* does not appear as a visual object. In this case, the graphical representation is done by the classes that inherits from it that includes *Statement*, *Fact*, and *PredicateDecomposition*. The shape of a *Statement* may be linked to a *ContextExpression* to indicate that it is the root of the analysis. The shape of a *Statement* contains the identifier and a text to describe the element. The *Statement* may receive the links from *Facts* or *PredicateDecomposition*. The *Fact* is represented in a similar way as a *Statement*, it has a shape with the identifier and description, but with a different format. *Fact* may be connected to a *ContextExpression*, *Statement* or *PredicateDecomposition* by a link, where the *Fact* always is the target. In the context diagrams *Facts* just appear in the end of decompositions.

The *Facts* are associated to *Variables* that describe the monitorable data that may be observed during the process configuration. The *Variable* class does not have a graphical representation. In both concrete syntaxes the *Variables* are represented as properties of *Variant*.

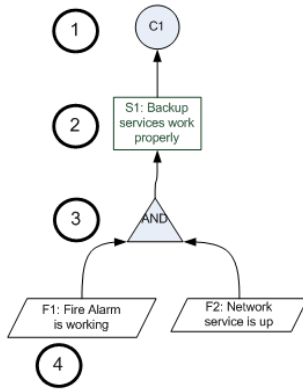
When it is necessary to decompose the expression it is done by the *PredicateDecomposition*. Hence, it has an attribute that indicates the type of decomposition (*AndDecomposition*, *OrDecomposition*, and *NotDecomposition*). The *PredicateDecomposition* may have several links arriving but only one leaving the shape. The decomposed element is the target of the link that departures from *PredicateDecomposition*.

Figure 3.13 describes the context model presenting the abstract syntax (a), the graphical concrete syntax (b), and the EMF reflexive editor concrete syntax (c). The numbers highlights the *BVCCoNContext* (1), the *Statement* (2) represented by the *S1: Backup services work properly* shape, a *PredicateDecomposition* (3) with an operator AND, and the *Facts* (4) with the description *F1: Fire Alarm is Working*. The decomposition of the context expression takes ad of the EMF reflexive editor presentation. Observe that the decomposition of the graphical concrete syntax (b) is equivalent to the tree-like structure of EMF reflexive editor concrete syntax (c).

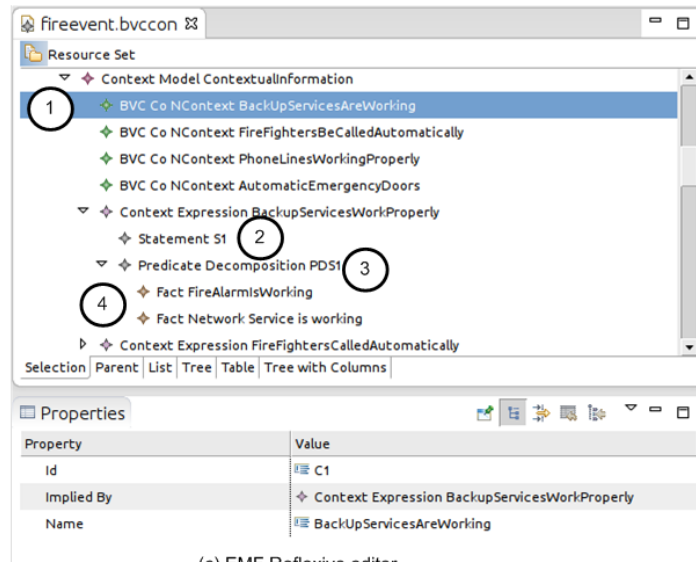
In order to associate the *BVCCoNContext* to a *Variant* we propose to



(a) Abstract Syntax



(b) Graphical Concrete Syntax



(c) EMF Reflexive editor
Concrete Syntax

Figure 3.13: Concrete Syntax for Context

hide the context analysis and link the *Variant* just with the higher level *BVCCoNContext*. The representation is done by connecting a *Variant* to a shape with the identifier of the *BVCCoNContext*, and the context shape is connected to a *VariationPoint*. If several *BVCCoNContexts* affect a *Variant* in that *VariationPoint*, then several connection between *Variant* and the *VariationPoint* are placed with the context shape in the middle.

3.3 Summary

In this chapter we presented the model and metamodel to support our approach. The metamodels describe both the abstract syntax and concrete syntax of the model required by our approach. The abstract syntax represents the necessary concepts and how they are related with each other. The metamodel also describes the constraint in OCL that helps to formalize the well-formedness rules. In the concrete syntax we present the diagrammatic elements necessary to create a model based on the abstract syntax definitions.

In several points of the definition we mention the BPMN language and link it to our metamodel. We employed an already validate metamodel that has well define its own abstract and concrete syntax. In Section 3.1.1 we present an excerpt of this metamodel, however, we will not detail it.

Both NFR and Context awareness metamodels were based on techniques presented in the literature: Chung et al. (2000) and Ali et al. (2010). We mapped the information about these models to a metamodeling language in order to allow the integration with our approach.

In the case of NFR there is a metamodel written in Telos (Mylopoulos et al. 1992), used for example in the OME 3 modeling tool. We based our representation of NFR in this metamodel, and performed the adaptations necessary to address our needs.

On the other hand, the contextualization technique presented by Ali et al. (2010) did not have a metamodel that could be translated, in this case we evaluated the literature available and designed a metamodel for their concepts.

Chapter 4

Configuring Business Process Models with NFR and Contexts

Dynamic environments require business processes to help achieving flexibility. This means that the process behavior must be able to change when it is necessary. In general, business processes create value to the organization by supporting the business. However, they may become invalid when external influences require a new process. Current BPM techniques are able to support changes such as process improvement and process re-engineering. However, an interruption of the process must be followed by a deep analysis of what could be changed. If the processes are flexible the problem with interruption may be reduced, since the business process may be designed from the beginning to change. The flexibility may be achieved by describing the process variability and configuring it.

Hence, the configuration of variability is an important step to achieve flexibility. By describing variability in the process we provide a means to obtain consistent processes that will keep track of the changes. After the description of the variability, the next problem is how to choose the best configuration for the situation. Since the stakeholders that know the business domain may have preferences, it would be interesting to model this preferences as well. Expressing this knowledge, in terms of quality attributes, would help to make the selection of process variants closer to the stakeholders' choices, even when they are not in charge of the configuration itself. We claim that Non-Functional Requirement (NFR) may be used to describe the quality attributes in a structured and flexible way. Thus, guiding the Business Process configuration using NFR may contribute to have a better fit of business process to the stakeholders' preferences driven by quality attributes.

Using stakeholders' preferences as guiding parameters for process changes results in business process closer to their expectations. However, the information that triggers the changes still must be explicitly described. When we are able to model and monitor the external influencing factors (i.e., contextual information), they could also be used to select valid variations (instances) for a specific setting. The inclusion of NFRs and contextual information as guiding forces of business process models configuration is our solution to achieve flexibility in business process models for dynamic environments.

In this chapter we present a configuration process called BVCCoN, which process describes activities that range from variability elicitation to the configuration of new business processes. We use a running example to illustrate each part of the approach. We start presenting the general process and then detail each step. We also present current implementation issues and conclude the chapter with a summary.

4.1 BVCCoN Process

The BVCCoN process is a description of tasks and resources that are part of our configuration process. The process presented in Figure 4.1 is composed of the following steps: *Elicit Variability*, *Describe Variability*, *Analyze Context*, *Links NFR & Variants* and *Perform Configuration*. The first four steps are performed at design time, while the last one, *Perform Configuration* step, at run-time.

The process starts with two sources of information: the initial business process models that will be the basis for the *Elicit Variability* step and the catalogs with NFR used in the *Links NFR & Variants* step. The other artifacts presented in Figure 4.1 will be produced by our approach. In the following sections we describe what will be used (input) and produced (output) for each step.

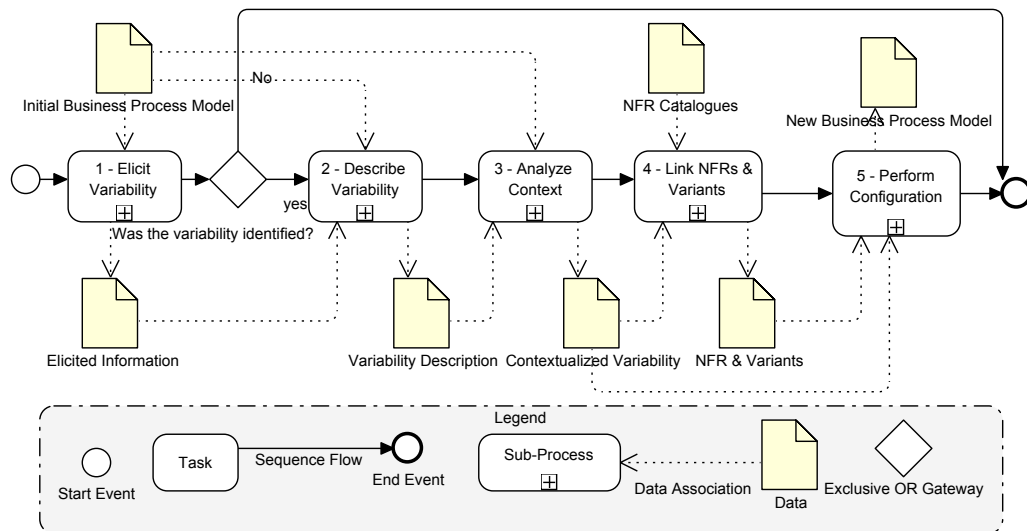


Figure 4.1: BVCCoN process overview

4.2 Variability Elicitation

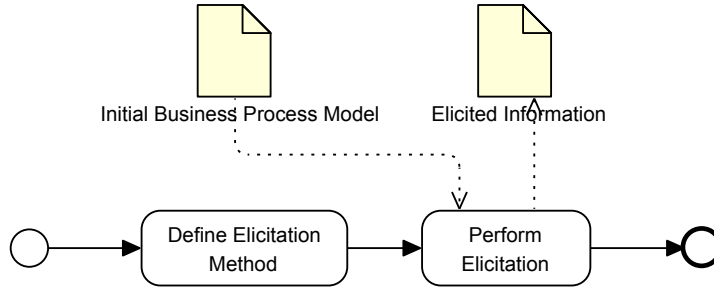


Figure 4.2: Variability Elicitation

The starting point of our approach is the variability acquisition of business process models, which requires information about process models and the business information where the process is performed. In this step we employ a requirement elicitation technique to obtain information about business process (see Figure 4.2). In the Requirement Engineering area there are several techniques to elicit the information necessary to describe variability (Kotonya & Sommerville 1998) such as interviews, documentation analysis, ethnography, etc. The elicitation of variability is the activity of identifying and discovering possible variations in a model. The objective is to identify different ways to carry out a process, the effects of the inclusion, change or exclusion of elements of the model. In this section we use a running example, the Fire disaster, to illustrate the steps.

Since several techniques can be used to elicit variability the condition to select a method is the ability of the method to extract useful information from the process model perspective. Take for example the information analysis framework proposed by Liaskos et al. (2006) that explores different facets of the information to obtain new data about it. According Liaskos et al. (2006) several information may be related to a goal. Based on a linguistic theory they describe facets that may drive the elicitation such as Agentive (Who will do), Dative (Who will be affected by), Objective (What is affected by), Factive (What are the results of), Process (How it will be done), Location (Where it will be done), Temporal (When it will be done), Conditional (What are the conditions to) and Extent (What are the degrees of). We adapted this framework for tasks of business process models providing a set of questions to drive the elicitation. Note that this is an initial set of questions, they need to be refined, excluded or complemented if necessary.

The facets that can be identified, with the respective questions, are:

- Agentive (Who will perform the task?)

- Dative (Who will be affected by the task?)
- Objective (What are the objects consumed or produced by the task?)
- Extent (To what degree the task will be performed?)
- Process (How will the action be executed?)
- Conditional (In what conditions will the task be performed?)
- Locational (Where will the task be performed?)
- Temporal (When will the task be performed?)

In the context of BPMN models we will use this framework to inquire about the tasks, activities and sub-processes of model and identify new information about them. The use of this framework is very simple as it is based on questions such as Who? How? When?, which are common place in requirements engineering elicitation techniques.

For example, in the context of a fire disaster the *Look for Smoke or Fire* task can result in the following answers for the questions. The agent responsible for performing the task could be a *Person* or a *Sensor* that will detect the smoke or fire. Some questions may not be applicable or have just one answer according to the context, e.g., the location where it happens.

- Who will perform the task?
 - Person
 - Sensor
- Who will be affected by the task?
 - Person
 - Ambient Intelligence System
- How will the action be executed?
 - Visual confirmation
 - Signal from a sensor
- Where will the task be performed?
 - In the building

The information will be structured according the information facet that originated the response. Thus, we can separate the agents of a task from the one that will be affected by it. The separation of these concerns contributes to a clearer identification of variability. For instance, if we were not able to identify a different place where the task *Look for smoke or fire* could be performed, it could indicate that there is no variation in this case. Indeed, it does not make sense identify fire in other building far from the one that the process was designed for.

Even with the support of a method to drive the elicitation there is an essential point in the elicitation process that must be carefully studied: the information source. We consider essential to obtain information from reliable sources such as experts in the business area, regulations, policies and documentation of the area. In the example of a Fire event we relied on documentation of procedures for fire events to acquire the information about the process (Raines 2011). This enabled us to identify the vocabulary of this domain and answer the questions about this process.

In order to obtain richer variability information it is recommended the use of several methods. They may be combined to obtain information and then structure the information to be usable in the next step of BVCCoN process. In fact, the questions that we derived from Liaskos et al. (2006) method could be combined with other acquisition methods such as classical interviews. A structured interview could be driven by these questions to obtain responses according to the facets of our structured approach. We also could do the same with a documentation analysis technique and drive it with a question set. The set itself could be generic enough to be reused in several business domains. Hence, our approach is different from other questionnaire methods (La Rosa et al. 2009) that must be reevaluated for every business domain.

The resulting information obtained may reveal concerns related with contexts where the tasks are performed. More specifically, the locational, temporal, and conditional facets may show information about the environment that surrounds the process. In further steps these information may be relevant for the codification of contexts during contextual analysis.

Once the Elicitation starts it may be repeated until the analyst believes that there is enough information to start the next step of BVCCoN process. However, this step may also show extreme cases where there is no variability. If the elicitation does not show any kind of variability two reasons could be highlighted: (i) the process runs in a very static environment so it does not requires to be configured; or (ii) the variability elicitation was not enough to find variability. Since our approach is focused on the configuration of variability, in the former case we recommend to stop the BVCCoN process and look for other solutions in literature. For the latter case, it could be

helpful to obtain more material to be analyzed from other sources.

The information elicited from the *Elicit Variability* step will be the input for the next step where they are analyzed and organized in the variability description.

4.3 Variability Description

The elicitation of variability results in information that must be analyzed to identify what may be modeled as variation points and variants. Among the information collected with the elicitation step there are clear cases of variability, as well as other cases that require further analysis. In this section we start to define a BVCCoN model (see Chapter 3). It will be constructed incrementally through the next sections. Hence, at the end of the chapter we will have illustrated the process using the Fire emergency running example.

Figure 4.3 outlines the description the variability in terms of variants and variation points. In this sub-process the first step is to identify where in the reference process there are variation points.

Then, the variation points and variants need to be linked. Observe that in order to promote the reuse in this step we also suggest the use of workflow patterns which may be associated to variation points. It is also important to carefully analyze the dependencies among variants.

During the elicitation information about task and its variations may be identified indicating the need to have variation points it may also be the case that a variation point may contain more than one task. As defined previously, the Variation Points are subject of changes represented by points in the business process work-flow model that may present alternative or variable ways to perform activities in the process. The first step in Figure 4.3, namely *Define Variation Points*, receives as input the reference business process model and elicited information. Based on this input the analyst defines the points to be marked as Variation Point and the tasks that will be part of it.

Since a business process work-flow needs to preserve the sequence of tasks, the analyst also must define where the Variation Point starts (*begins*) and finishes (*ends*). This information is required to perform changes during the configuration. The output of this step is the reference process model marked with variation points, which is part of *Variability Description* (see Figure 4.3).

Figure 4.4 which describes a possible reference model considered for Fire emergency running example. We identify four variation points: VP1 associated to *Look for Smoke or Fire* task; VP2 related to *Sound Fire Alarm* task;

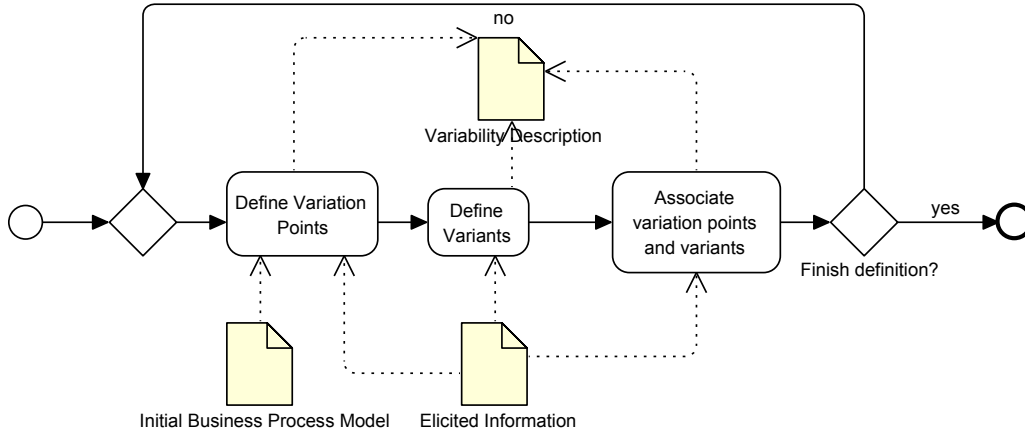


Figure 4.3: Variability description step

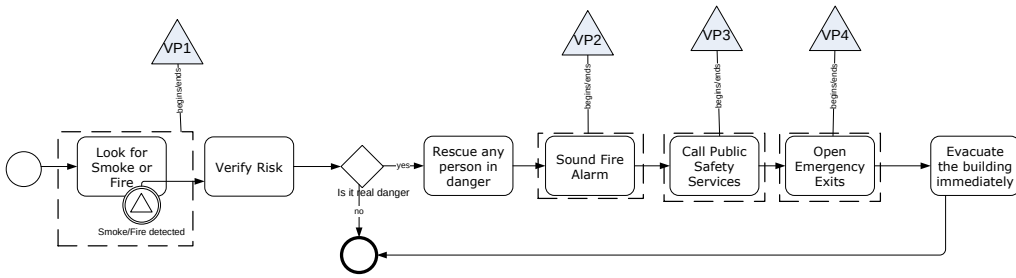


Figure 4.4: Example of reference model with variation points

VP3 corresponds to *Call Public Safety Services* task; and VP4 is linked to *Open Emergency Exit* task. Note that the first variation point encloses more than one task while the other are linked to just one task. After defining the variation points and selecting where they begin and end the next step is to define the elements that become part of the variants.

The information acquired during elicitation and the previously defined variation points are the inputs for *Define Variants* step (see Figure 4.3). The variants are objects of change representing alternative or variable ways as the process activities may be performed. Thus, they are associated with BPMN fragments that will express the way a process may be modified to adapt to new environment settings.

Similarly to the variation points, the variants are defined by the analyst. Hence, the expert will be responsible for the definition of BPMN fragments that correspond to the variation point. It is expected that the information collected during elicitation is sufficient to identify variants. In general, if

the elicitation indicates more than one behavior for the same task they may become variants.

In the Fire emergency event example we identified variants that are associated to the type of agent that will perform some tasks. This is the case with *Look for Fire and Smoke* task, which may be performed by a person or a sensor. In this case, two variants were described a manual task to represent a person doing the action, as well as a service task for the case a sensor is used. Variation also happens when we associate variants and variation points as will be described in next section.

The variability description may also affect the way the tasks behave. The work-flow patterns (Wohed et al. 2005) describes behaviors of activities when associated to a variation point. In BVCCoN model, the application of work-flow patterns describes how to combine the variants and variation points in a controlled way. In the variability description step (see Figure 4.3) the last activity is the *Associate Variation Points and Variants* task.

The variation points include operators to link them to the variants. Note that many variability models use the (Pohl et al. 2005) mandatory, optional, and alternative terms to indicate how the elements may be associated. In our case we use AND, OR, and XOR terms, which corresponds respectively to the standard interpretation of literature. When the AND operator is associated to the variants it means that all variants must be part of the solution. The OR operator allows the representation of options where one or more variant may be part of the solution. Last but not least the XOR operator represents the alternative behaviors, when just one of the variants for that variation point will be part of the process models.

The variation points' operators influence the workflow patterns that are associated to variants. Consider for instance the AND operator, allow two patterns may include in the solution, the *sequence* pattern and the *parallel split/synchronization* pattern. Note that in both cases all variants associated with an AND variation point should be part of the solution. Hence, the work-flow patterns define if they will be in sequence or in parallel. On the other hand, XOR and OR operators may be associated to two patterns, the *Exclusive Choice* and *Multiple Choice* respectively.

Figure 4.5 represents the variants associated with the variation points for the *Look for Smoke or Fire* task with a manual and an automatic alternatives. Other variants also are associated with the manual and automatic choices. For example, in the VP2 variation point allows the *Sound alarm manually* and *Sound alarm automatically* variants. The VP3 variation point that points to the *Call Public Safety Services* task assumes three alternatives: automatically, manually by phone line, or manually by a mobile phone. Observe that these three ways to call safety services (e.g., firefighters) are mutu-

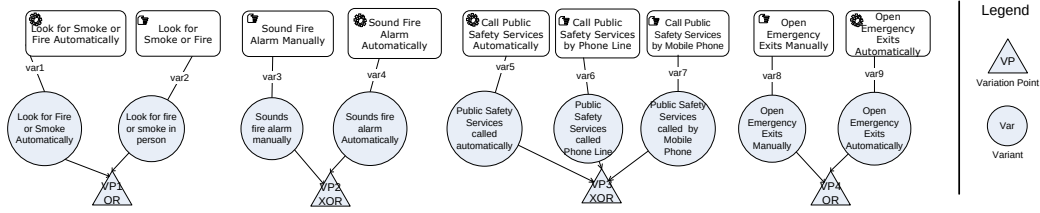


Figure 4.5: Example of Variants and Variation points

ally exclusive. On the other hand, the VP4 variation point may accept both *Open Emergency Exits Manually* and *Open Emergency Exits Automatically* variants at once.

As a result of the variability description an extended model rich with variability information is produced (see Figure 4.3). This model is the input for the next steps and will be linked to other models to support the configuration mechanism. Having concluded the variability description the next step is to focus on contextual information.

4.4 Context Analysis

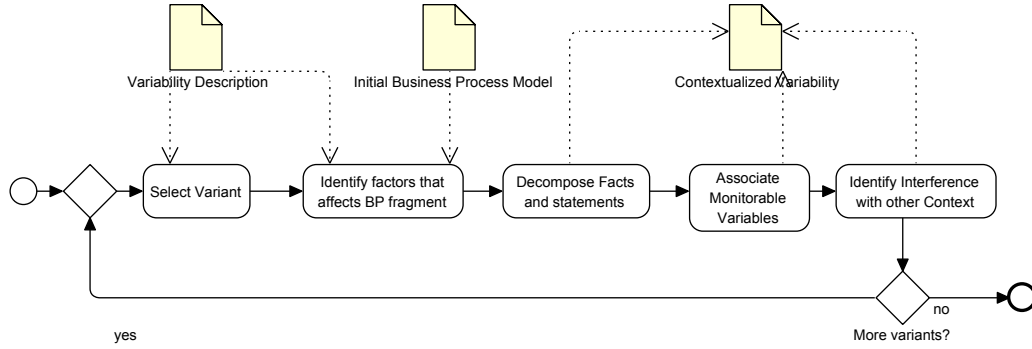


Figure 4.6: Context Analysis

A context can be defined as a state of the world that is relevant to an actor goal (Ali et al. 2010). In this step the business process model is analyzed to identify the contexts that could affect the model. Note that contexts can be identified by examining the domain and the relationship of the actors and domain concepts (Bai et al. 2008). The user or system states in the environment are described as contexts. The context can describe *What is going on?*, *Where are they located?*, *What are the resources available for*

use?, and so on. The context is composed of expressions/variables that need to be evaluated to check if the context holds.

The context analysis reviews information obtained during elicitation of variability in order to check whether they may lead to changes in the way that the process performs. The contexts can be analyzed using a decomposition of facts and statements that are connect to them (Ali et al. 2010). The decomposition ends when it is possible to describe all variables that allow the verification if the context holds. Hence, the goal is to obtain a formula of facts with all facts supported by monitorable variables. Once defined, we can identify the relationship among contexts, variants and variation points. The output of the context analysis is a contextualized variability description. Figure 4.6 represent the context analysis step. In order to describe the contexts the following steps are required:

1. Select a variant of the variability model to be evaluated;
2. Identify factors that affect the execution of the business process fragment of the given variant;
3. Decompose facts and statements that support the context;
4. Associate monitorable variables to the facts;
5. Validate the interference with other context;
6. Repeat the steps for other variants.

At the end of these steps the analyst will have a set of contexts associated with the variants. Note that context may interfere with other contexts. Hence, it is necessary to evaluate the interference among all the contexts (see Figure 4.6). The result of this evaluation may reduce the number of contexts and how they interact with the variants.

In the following we use the Fire emergency example to demonstrate context analysis step. Recall that this running example is a classic case of a socio-technical system where both human and software should perform safely. This system has a critical task that keeps the environment monitored and alerts when necessary. Thus, in several points of the process there are interferences of the environment in the way how the process may be executed.

After a careful analysis we identified some situations when the system required performing some action in the scenario and becomes vulnerable to fail. For example, the sensors and the infrastructure that support the software system should allow safe execution of tasks. Otherwise, the human intervention could be necessary to correct failures of the software system.

Recall that in our proposal we associate the contexts to variants instead of specific process elements (De La Vara et al. 2010a). The reason is that a context may affect several tasks at once. However, in order to avoid the description of a variation point without valid variants, the analyst has to identify at least one valid variant in every situation. Hence, a logical evaluation of contexts is required in order to find a situation where at least a context is true. Otherwise, a pragmatic solution would have the variant without a context annotation. The latter allows a default solution in the case of all contexts became invalid. For example, the *Sound Fire Alarm Manually* would be a default solution in the case an automatic alarm is not working.

In our running example we identified that *Call Public Safety Services* task and its variants may be affected by contexts. First, the context about the methods to call safety services must be valid. Initially the software system that monitor and automatically alerts the safety services must be working properly. If it is not, the alternative is to use manual methods such make a call using a phone line or a mobile phone to alert public safety service (e.g. Fire department). However, both are only applicable in the context it is favorable, i.e. the phone lines are working and the risk to stay inside the building is not high.

Figure 4.7 shows a decomposition of the context as explained above. The *Fire Fighters Called Automatically* context is supported by three facts: *Fire alarms are active*, *Fire was confirmed*, and *Network is working*. When the variables associated with this context formula match the specified values, the context holds and the variant *Call Public Safety Services Automatically* may be part of the process.

Observe that in the case of change in the context values all pairs of variation points and variant must be re-evaluated to determine the most preferred solution. For example, if the *Fire Fighters Called Automatically* context becomes invalid, the *Call Public Safety Services* task cannot be performed automatically anymore and an alternative task shall be considered.

As a result of this step the contextualized variability information is produced. The addition of contextual information enriches the BVCCoN model with the information necessary to trigger the changes during configuration. Hence, only valid variants will be evaluated during the selection of a new process. However, the selection of suitable process fragment is also dependent on some Non-Functional Information, that is, quality attributes may be helpful and need to be linked to variants.

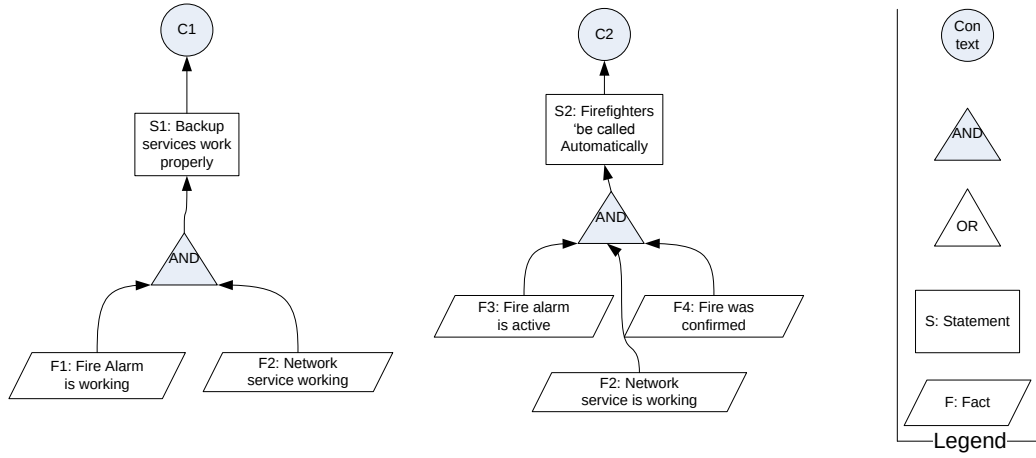


Figure 4.7: Example of context decomposition

4.5 Link variants & NFR

In this section, we explain the use of NFR in BVCCoN approach. During the business process configuration it is important that the process reflects the stakeholders' needs. In general, the stakeholders state their preferences during the configuration of process (La Rosa 2009). This practice has a limited effect because the stakeholders that helped the analyst during the design of the process may be unavailable during the configuration. Moreover, dynamic configuration requires fast responses, which may not be possible if human intervention is required. In our approach we propose to store stakeholders' preferences using NFRs, recall that NFRs are defined as requirements that describe the properties, characteristics or constraints, as well as quality attributes that a system must have (Mairiza et al. 2010). In doing so during configuration the selection of variants may be performed without the presence of stakeholders.

In this step we identify the NFRs that are important for the process and define the impact of each variant on the NFR. The NFRs are linked to the business process variants earlier identified. Figure 4.8 shows an overview of this step with the associated tasks and resources.

We begin identifying which NFRs will be taken into consideration. The output of this step is a description of the NFRs and Variants. This NFR elicitation may be performed using any suitable technique. For example, interviewing people involved in the business process (Aburub et al. 2007), using requirements catalogs (Cardoso et al. 2009) or relying on a mix of elicitation techniques.

We now can associate NFRs with variants to represent preferences of

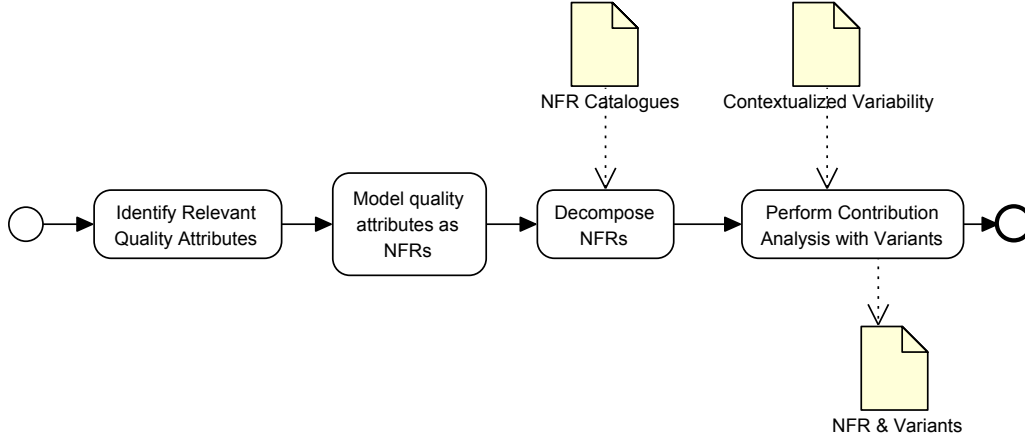


Figure 4.8: NFR analysis

stakeholders over the possible variant selection. NFRs will represent quality attributes that the stakeholders expect to be met by the system.

It is important to remember that we are considering system as being composed by software, persons, and processes (Maté & Silva 2005). Thus, quality attributes of software, i.e. NFRs, may affect the process as well. The analyst that is modeling the stakeholders' preferences has the freedom to include or remove NFRs from the list NFR to be evaluated. Indeed, there are cases where the domain requires some specific quality attributes that may not be captured in the available catalogs. Once the NFRs are identified, we perform the linkage between the process variants and the requirements.

Let us consider the qualitative scale of the NFR Framework proposed by Chung et al. (2000). In their framework, the most positive impact on a non-functional requirement is called *Make*, a partial positive impact is named *Help*, a partial negative impact is assigned *Hurt* and the most negative impact is denominated *Break*. These values can be mapped, respectively, to ++, +, − and −−, in our scale. In order to make it more user friendly, this scale can be replaced by any other suitable scale. Figure 4.9 shows a partial view of BVCCoN model for the running example. Observe that the NFRs are related to variants through the contribution links.

In our running example, we selected two NFRs for consideration: *Performance* that indicates how well the variants will contribute to an immediate response; and *Reliability* that captures how reliable the variants are for this situation. Every solution that automatically performs a task or increases the parallelism will contribute positively to *Performance* quality attribute, since less time will be spent in these tasks. However, the automated activities rely on an infrastructure that can be affected by the disaster and become

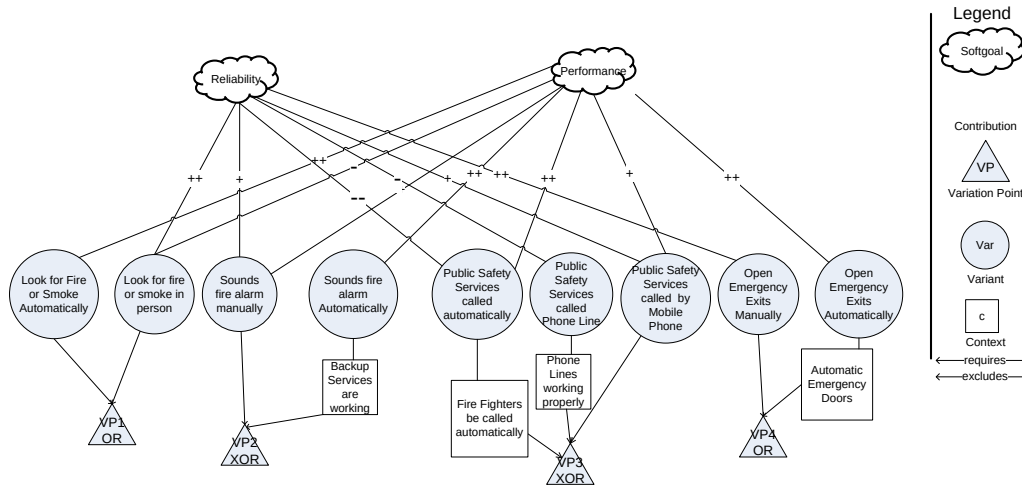


Figure 4.9: Use of a NFR Catalogue to select variants

unavailable, which can reduce the *Reliability* of the process as a whole.

The best example to illustrate the various possibility is consider the *Call Public Safety Services* activity (e.g., *Firefighters*). In Table 4.1 we present the contribution for the *Reliability* and *Performance* NFRs from the possible variants of the *Call Public Safety Service* task. If performed automatically by an *Ambient Intelligence System* it will have a fast response. However, it depends on the network infrastructure that can be disabled in a more critical disaster. On the other hand, using a *phone line* could delay the response but it is more reliable since there are fixed phones (and mobile phones) replicated in several places which can be less affected by some disasters.

Table 4.1: Variants and their relationships with Non-functional requirements

ID	Variants	NFRs	
		Reliability	Performance
var1	Look for Fire or Smoke in Person	++	—
var2	Look for Fire or Smoke Automatically		++
var3	Sounds fire alarm manually	+	+
var4	Sounds fire alarm Automatically		++
var5	Public Safety called automatically	--	++
var6	Public Safety called from Phone Line	—	
var7	Public Safety called by Mobile phone	+	+
var8	Open Emergency Exits Manually	++	
var9	Open Emergency Exits Automatically		++

The NFR can be used to prioritize the variants thus helping with the selection of the best configuration. Since many alternatives can emerge during the elicitation process, the contribution analysis can be time consuming. By including contexts the complexity of the configuration of business process can become quite difficult, since it is now necessary to consider the interaction between contexts and variants. However, using NFRs as criteria can contribute to reduce the variability space, i.e., it can help to drive the modification process.

In this step we identify the NFRs that are important for the process, then we define the impact of each Variant to the NFR by means of the contribution link. This information can be gathered interviewing people involved in the business process, using requirements catalogues or any mix of elicitation techniques. Note that the NFR analysis can indicate that several non-functional requirements have to be met simultaneously.

4.6 Perform the configuration

The previous step generated an extended model to be used during configuration. Recall that the initial model was based on evaluation of a reference model, which as gradually enhanced to include a description of possible variability and contextual information. As well as, the representation of quality attributes which captures stakeholders' preferences. In this last step of BVC-CoN approach a configuration is performed generating a new business process model as output.

In this step we consider the Variation Points and the Variants of the business process, and how they impact the non-functional requirements. This information can be used to support the configuration itself. It can be performed based on Variants selection or the most critical NFRs.

Figure 4.10 details the activities of this *Perform Configuration* sub-process. The first step is to define the NFRs priorities. In this thesis we focus on the process configuration based on prioritization. Note that some solutions can rely solely on expert judgment and NFRs to resolve conflict at design time (Chung et al. 2000, Horkoff & Yu 2010). However, since we are dealing with runtime adaptability, it may not be possible to rely on experts (e.g., they could not be available). Thus, the definition of priority allows solve potential conflicts at runtime. Moreover, we require the NFR prioritization to be conducted before entering in the *Context Monitoring* loop (see Figure 4.10). The prioritization is performed by an analyst, who gives weight to each NFR. In our proposal NFRs with higher priority have higher weights.

After the NFR prioritization the next step is the *Context Monitoring* loop

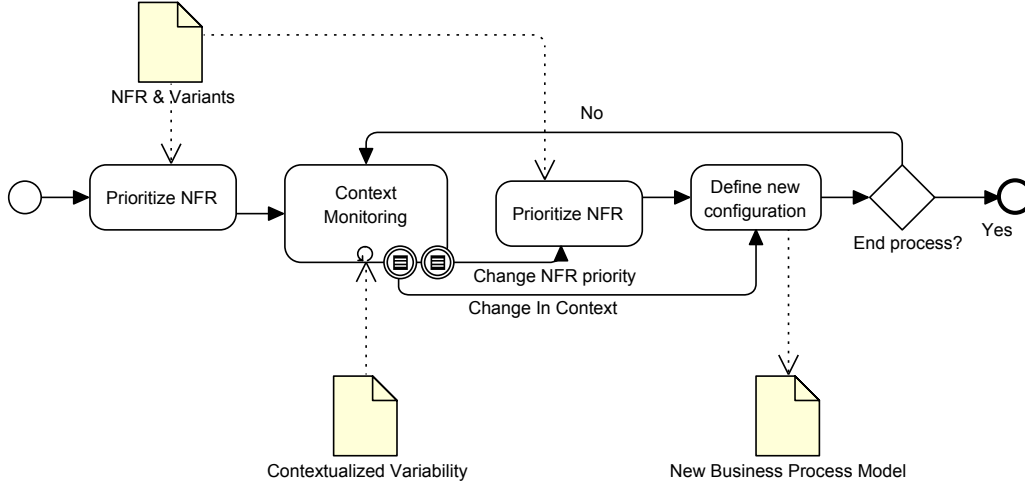


Figure 4.10: Perform Configuration sub-process

(see Figure 4.10). The contexts defined in our model will be monitored to identify changes in the context variables. These variables are associated to sensors, user inputs and other information sources that may trigger contextual changes. Once the variable values change the contexts are evaluated and new configuration must be defined.

In our proposal we identify two ways to exit the *Context Monitoring* loop, namely, changes in the contexts values or changes in NFRs priorities. In the former, we apply the typical context evaluation to trigger the definition of new a configuration. Recall that the NFRs priorities are defined by an analyst but are not part of the context model. In order to deal with changes in the NFRs priorities we include another prioritization step in the process (see Figure 4.10). Although our approach aims to reduce the human intervention during the configuration, this step is an exception to give to the analyst the chance to update priorities over NFRs.

Finally, the *Define new configuration* task is performed once the priorities are set and changes in context values are identified. The rest of this section will describe the algorithms used and present how a new configuration is produced.

One way to compute the variants priorities is to make a weighted average over NFRs (Ali et al. 2010). In this case, contributions can be counted and weighted according to the NFRs preferences. This method has limitations to deal with NFRs since it hide information about the contributions strength. For example, using weighted average the number of negative contribution could null the effect of the positive ones.

In order to obtain a global ranking that takes into account the local inter-

actions, we adopted a multi-criteria decision making method. We chose the Analytic Hierarchy Process (AHP) (Saaty 2008) method which generates a global preference measure based on the choice among alternatives. The AHP was selected because it fits well with the structures used in our approach (see Figure 4.9). For example, the AHP's hierarchical criteria may be represented by the NFR decomposition. The preferences among alternatives and criteria is represented by the contributions of Variants to NFRs. Moreover, the use of priorities over NFR is also easily taken in account by the AHP.

The selection algorithm is described in Figure 4.11. Let us consider N as the set of NFR, Var as the set of Variants, VP as the set of Variation Points, C as the set of Contexts, and $contrib(v, n)$: the value of the contribution link from the variant v to the NFR n . The contribution function varies in the range: $1 \geq contrib \geq -1$. The NFR have weights (w) associated to them to express their different priorities.

```

1: for all  $c \in C$  do
2:   validate( $c$ )
3: end for
4: for all  $vp \in VP$  do
5:   for all  $v_i \in Var$ , where  $v_i$  is part of  $vp$  do
6:     if isValid( $v_i$ ) then
7:       validVar[ $i$ ]  $\leftarrow v_i$ 
8:       for all  $n_j \in N$  do
9:         contribMatrix[ $i$ ][ $j$ ]  $\leftarrow contrib(v_i, n_j)$ 
10:      end for
11:    end if
12:  end for
13: end for
14: ranking[]  $\leftarrow$  AHP(contribMatrix[], validVar[],  $w$ )
15: solve( $VP$ , ranking[])

```

Figure 4.11: Select new configuration algorithm.

First, the algorithm computes the valid contexts (lines 1-3). Considering our Fire emergency running example, if the *FireAlarmOn* context expression is computed as true then the *FireAlarmOn* context is valid.

After that, the Variation Points are evaluated to identify the valid Variants (lines 4-6). There are two ways to be valid: being associated to a valid context, or having no context associated to it (default situation). The valid Variants are included in a specific set (line 7), and their contributions to the NFR are computed (lines 8-10). Since, we do not deal with manual selection

the user is not able to interfere in the selection of variants. The set of valid Variants and their contributions to NFR are the input for the AHP method (line 14) that will compute a global ranking of preference among Variants. In its turn, this ranking is the input to a solver that will derive a valid solution considering the relationships (i.e., exclude and include) among the Variants.

At this point we know the Variation Points and the Variants of the business process, and how they impact the Non-Functional Requirements. Now we will use this information to support the configuration itself. There are various possible ways for analyzing the impact of each configuration on the NFR: for example, top-down analysis or bottom-up analysis. In the top-down analysis we select which non-functional requirement has the maximum priority, and then derive a process configuration that maximizes the selected NFR. Alternatively, in the bottom-up analysis we define a process configuration, by selecting a subset of variants, and then observe how this configuration affects the non-functional requirements. As previously presented, in this thesis we adopted the top-down analysis leaving the bottom-up analysis as future work.

Once the set of Variants are selected they are grouped in a new instance of the business process model. Each Variant has its pattern evaluated (see Sections 2.1.5 and 3.1.2) and the appropriated action is selected. Figure 4.12 presents a solution process for prioritization of the *Performance* NFR. Observe that the selected variants were the ones that include the automatic execution of the activity, represented by a gear on the top of the task. It was selected by analyzing the NFRs contributions. Since we are using contexts the process can dynamically change and be reconfigured at run-time to adapt to the situation.

Recall that in order to support our approach we designed several meta-models in Chapter 3 and the configuration algorithm in this section.

In the last step, *Perform Configuration*, the approach uses model transformations in order to generate a new process model. It could be argued that since the process transformations are performed with a known number of variants, the performance could be improved by executing the transformations before the last step. Indeed, it make sense in small examples as the presented in this Chapter. However, the exponential growth of the number of solutions generated may impact the storage of process. For instance, if the model has three variation points with three variants each, the potential the number of resulting process models are not nine but 27 variants. In the same way, a model with larger number of variants could require several times more space for storage. We believe that the approach as-is trades the performance, in the last step, for maintainability, since the number of possible solutions is controlled by the transformations mechanism. On the other hand, the use of

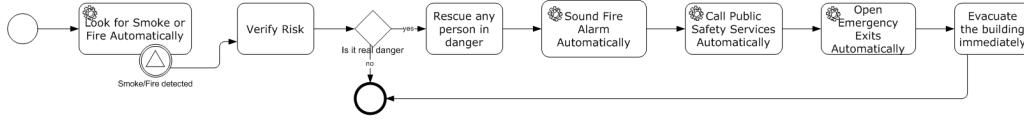


Figure 4.12: Example of process instance, the AmI System performing some tasks.

pre-compiled models could be further researched in future works.

In next section, we discuss about the implementation of this algorithm and the use of the meta-model to support model transformations that will generate the business process model.

4.7 Implementation Issues

The BVCCoN approach was designed in two parts with a description of concepts through a meta-model and the use of these concepts during the configuration process. In this section we describe how we implemented the algorithm of Figure 4.11. We also detail how the NFR structure is transformed into an AHP analysis table. Moreover, we describe the model transformations used to create new process. At the moment, the implementation is not completely supported by a tool. Hence, in the sequel we outline the implementation as proof of concept. In doing so we hope to demonstrate the technical viability of BVCCoN approach.

4.7.1 Configuration Algorithm

In Chapter 3 we described a meta-model of BVCCoN approach that uses as base the Eclipse EMF (Eclipse.org 2012) framework. Among the capabilities of the Eclipse modeling tools there is the possibility to generate editors for the models designed in EMF. These editors may be generated by using several techniques such as models transformation (e.g., GMF) or graphical design of models elements (e.g., Graphiti Framework).

The use of a meta-model based editor allows the validation the models using OCL language (OMG 2010b). This will enable the syntactic validation of the model. As presented in Chapter 3, the support for OCL allows validate models before using it other processing mechanisms such as the configuration algorithm or the model transformations. Thus, fails related to syntactic consistence are reduced during execution of other steps of our approach.

We designed a simple editor using model transformation of EMF framework. It will produce an EMF tree-like reflexive editor instead of a full

graphical editor as specified in Chapter 3. However, the model produced by this editor will follow the well-formedness rules stated in the meta-model.

We analyzed the configuration algorithm (outlined in Figure 4.11) using the Big O asymptotic notation. Let's consider c the number of contexts, n the number of NFRs, v the number of variants, and m the number of variation points. According with these values the only value that will vary is the number of variants, because some of them will become invalid due contextual change. So we have $f(v) = |c + v(1+n) + nv^2 + (n)^2 + v \log v| = O(v^2)$, where the terms are c for evaluation of context, $v(1+n)$ validation of variants, $nv^2 + n^2$ the AHP analysis, and $v \log v$ the solving method. Hence, the algorithm has a complexity $O(v^2)$ order. A heuristic search can be used to solve it. Another alternative, is to use a SAT Solver. In this case the asymptotic complexity is as good as the SAT Solver that provides the solution.

4.7.2 AHP implementation

The stakeholders' preferences are expressed in terms of contributions from variants to NFRs. During configuration the algorithm uses that information to select the most preferred solution. In order to do so, it is necessary to transform the contribution graph into a matrix that allows the AHP method to work.

For each NFR a matrix is created that compare all values of contribution. The ++ to -- range is evaluated according to the importance in achieving the NFR. A matrix $n \times n$ (with n as the number of variants) will associate the variants according to their contribution. The scale used (Saaty 2008) varies from 1 (when they have the same importance) to 9 (when there is a big difference among the variants). In Table 1 we present the mapping used to convert from the NFRs contribution to the scale used in AHP. Note that in the first row ++ sign has a growing importance from 1 (when compared to ++) to 9 (when compared to --). On the other hand, in the last row the -- sign has an inverse importance varying from 1/9 (when compared to ++) to 1 (when compared to --).

For instance, in our running example the *Look for Fire or Smoke in Person* variant contributes with ++ to the Reliability and the *Look for Fire or Smoke in Automatically* variant does not influence the value (i.e., the same as = sign in our notation). Since ++ contribution is more important than a = to the Reliability it receives the 5 value. The Table 4.3 shows the comparison of all variants for this NFR (see table 4.1). The identifier var1 to var9 refer to the following variants: Look for Fire or Smoke in Person (var1); Look for Fire or Smoke Automatically (var2); Sounds fire alarm manually (var3); Sounds fire alarm Automatically (var4); Public Safety called auto-

Table 4.2: Mapping from NFRs Contributions to AHP values

$\rightarrow$	++	+	=	-	--
++	1	3	5	7	9
+	1/3	1	3	5	7
=	1/5	1/3	1	3	5
-	1/7	1/5	1/3	1	3
--	1/9	1/7	1/5	1/3	1

matically (var5); Public Safety called from Phone Line (var6); Public Safety called by Mobile phone (var7); Open Emergency Exits Manually (var8); and Open Emergency Exits Automatically (var9).

After summing and normalizing all values of a line, we obtain the total dominance of a variant values over a NFR. All this values are grouped in a new matrix. Hence, using the NFRs priorities we compute the global priority of each variant as a ranking. In this ranking each position indicates the global priority of an element over the next one.

4.7.3 Model transformation

The business process configuration involves the creation of new BPMN models. In the last step of BVCCoN process the configuration will produce the new process models. In order to implement the generation of new process models we used the model transformation techniques. The model transformations require the meta-models presented in Chapter 3. Hence, using the BVCCoN model as source we derive a BPMN model that corresponds to the selected variants.

Both the BVCCoN model and reference models are read and queried by the model transformation engine that select the transformation required to generate a new process model. Thus, the output will be a new BPMN model. The transformation select for each variation point will correspond to a transformation that matches the workflow pattern associated by the variation point operator. It is also necessary to query the BVCCoN model to confirm existence of alternative variants and default solutions.

Figure 4.13 shows a model transformation in QVTO (Query/Verification/Transformation Operational) language to apply a workflow pattern for the variants in a variation point. This is an excerpt of the transformation script that receives a BVCCoN model and BPMN models to produce a new BPMN model. In this excerpt we present the mapping for the Multiple Choice work-flow pattern. This transformation take as input a Variation Point and output a Process (line 1). The first part of transformation makes

Table 4.3: AHP Matrix for reliability

	<i>var1</i>	<i>var2</i>	<i>var3</i>	<i>var4</i>	<i>var5</i>	<i>var6</i>	<i>var7</i>	<i>var8</i>	<i>var9</i>
<i>var1</i>	1	5	3	5	9	7	3	1	5
<i>var2</i>	1/5	1	1/3	1	5	3	1/3	1/5	1
<i>var3</i>	1/3	3	1	3	7	5	1	1/3	3
<i>var4</i>	1/5	1	1/3	1	5	3	1/3	1/5	1
<i>var5</i>	1/9	1/5	1/7	1/5	1	1/3	1/7	1/9	1/5
<i>var6</i>	1/7	1/3	1/5	1/3	3	1	1/5	1/7	1/3
<i>var7</i>	1/3	3	1	3	7	5	1	1/3	3
<i>var8</i>	1	5	3	5	9	7	3	1	5
<i>var9</i>	1/5	1	1/3	1	5	3	1/3	1/5	1

some assessments about the type of *VariationPoint* and the workflow pattern that will be applied (lines 3-5). In this case the assessment test if the *VariationPoint* operator is the *OR* type, as well as, if the workflow pattern is the *MultipleChoice* type. There is also a test to verify if the *begins/ends* of the variation point are equals. When these assessments are valid the script starts by initializing the variables to be used (lines 7-9). After that the begins and ends points of a *VariationPoint* are stored in variables (lines 11-13). All elements (i.e., *FlowElements*) from the reference process marked by a *VariationPoint* are then removed (line 15). In the sequel, new gateways (*InclusiveGateway*) are initialized to represent the split and join from a multiple choice pattern (lines 19-22). Recall that the *Variants* are linked to business process fragments, in order to include them in the process model the *Variants* must be evaluated (line 24). Hence, the business process fragments of each Variant is linked to the split (lines 26-28) and join (line 29-31) gateways using *SequenceFlows* (lines 25 and 28).

The process model is accessed by the *result* variable (lines 8, 21-22, 25, 28). Note that the process model that will be produced is declared in the signature of the mapping (i.e., *Process* - line 1). After all the mapping actions are applied the model resulting will be compatible with BPMN 2.0 meta-model used by our proposal as presented in Chapter 3. Other mappings are available for each workflow patterns present in the BVCCoN approach (i.e., sequence, parallel, and exclusive choice). Moreover, the model transformation script also evaluates the BVCCoN model to select default solutions when necessary.

Consider the Fire emergency running example, after the application of the

```

1 mapping VariationPoint::applyMultiChoicePattern(): Process
2 when {
3   (self.operator = VariationPointOperator::OR) and
4   (self.variants.pattern.wfPattern->includes(WorkflowPattern::
5     MULTIPLE_CHOICE)) and
6   (self.begins <> self.ends) ){
7   init {
8     result := processOld;
9   }
10
11   var t : Set(FlowElement);
12   t += begins;
13   t += ends;
14
15   var be: Set(FlowElement) := processOld.flowElements - (t->
16     intersection(processOld.flowElements));
17
18   if (begins.ocllsUndefined() = false)
19   then {
20     var split = new InclusiveGateway();
21     var join = new InclusiveGateway();
22     result.flowElements += split;
23     result.flowElements += join;
24
25     variants->forEach(v){
26       var splitVarBeginSeq = new SequenceFlow();
27       splitVarBeginSeq.sourceRef:= split.oclAsType(FlowNode);
28       splitVarBeginSeq.targetRef:= v.part.begins.oclAsType(
29         FlowNode);
30       result.flowElements += splitVarBeginSeq;
31       var varEndJoinSeq = new SequenceFlow();
32       varEndJoinSeq.sourceRef:= v.part.ends.oclAsType(
33         FlowNode);
34       varEndJoinSeq.targetRef:= join.oclAsType(FlowNode);
35       result.flowElements += varEndJoinSeq;
36     }
37   }
38   endif;
39 }

```

Figure 4.13: Model transformation with QVTO.

multiple choice workflow pattern presented in Figure 4.13, the *Sound Fire Alarm* variation point in Figure 4.14 (A) is transformed into a new process. Figure 4.14 (B) presents an excerpt of the process model derived, where the elements marked by the variation point are replaced the *Sound Fire Alarm*

Automatically and Sound Fire Alarm Manually variants.

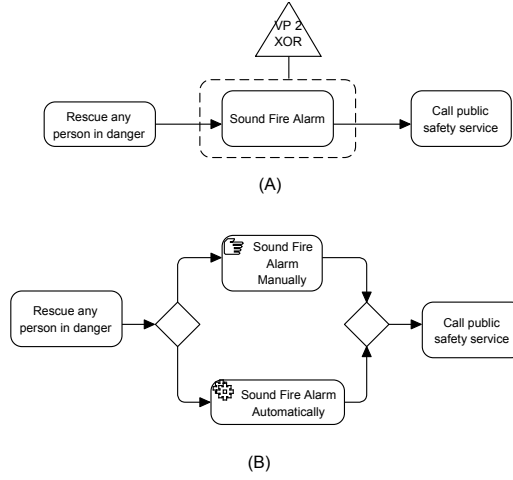


Figure 4.14: Example of Variation Point and Process

Even though we do not provide an integrated environment for monitoring and configuration. The process generated will be aligned with the NFRs' priorities as well as the contextual information. With that we demonstrate the technical viability of our approach. Next, we discuss the contribution presented in this chapter.

4.8 Summary

As part of our approach to achieve flexibility in business process models we presented in this chapter a process that supports the activities of the configuration. Our proposal covers from the elicitation of process variability to the generation of process variants adapted to specific settings.

The elicitation enforces the importance of the variability acquisition for our approach. In the example we applied a method of variability acquisition (Liaskos et al. 2006) that allows identify several aspects of the information that could become variants in the process. Even though this method is specific for variability, it is not unique and can be replaced by other methods such as interviews or documentation analysis. The condition to use other methods is the ability to separate what is variation from what is common. If the method allows this separation it could be used with our approach.

After the elicitation of variability the main problem is to structure the information as variants and variation points. Since we are dealing with process information, both are expressed as business process elements. It has impact over the way how the selection will be conducted. For instance, if the granularity of variants were controlled (i.e., just a task by variant) the logical description of the process variability could become very complex. As we can see in Montero et al. (2008a) using simple tasks as variant units requires complex structures of Feature Models and a sophisticated control algorithm. We avoid these problems but the counterpart is that the business analyst should pay attention to the coherence of their descriptions. If the business analyst fails in providing coherent description of process fragments our approach does not ensure the quality of the results. However, we believe that it is the same for any other variability description approach.

When considering the relationship among NFRs and contextual information in our approach we identify that it is not possible to express the impact of contextual information in the NFRs. In some specific case the context (e.g., security alert) may force changes in the NFRs priorities independent of the way how stakeholders stated during the link among NFRs and variants. This alignment may be done manually if necessary by changing the priority description. However, it would be more practical if we could do it automatically. Even so, we still believe that with some improvements our approach could overcome this limitation.

In next chapter we describe the assessment of our approach using controlled study based on simulations. With that assessment we expect to demonstrate that our approach works and show in what situations it is useful.

Chapter 5

Simulation Based Assessment

In this Chapter we present an investigation showing the design, preparation and results of the simulation of business processes models generated by BVCCoN approach. Business process models may describe several views of an organization processes (Aguilar-Savén 2004). They may cover from data description to behavioral models such as the activities flows from BPMN. Several tools may be used to evaluate these models, in special metrics and simulations.

It is worth noting that the deployment of a process without previous knowledge of its capabilities may expose the organizations to situations that increase risk and costs (Wernick & Hall 2007). Thus, using simulations it is possible to manage these problems in a controlled way and investigate the impact of a business process model.

In order to evaluate the proposal presented previously in Chapter 4 we designed a quantitative simulation based investigation. The objective of this Chapter is to assess the use BVCCoN approach in circumstances that it would be useful. Thus, we compare the use of business process models generated by our approach to a standard process that was not designed to be adaptable. We hope to demonstrate the importance of using contexts and NFRs to the process model configuration when compared to a static process model.

Several factors including resources availability, human interference and other aspects may change during the execution of process models (Rosemann et al. 2008). These factors may be modeled into process model or using external representation, when they are out the organization control zone.

In dynamic environments the information required by the process may change fast thus the processes must be able adapt accordingly. In our approach we rely on NFRs and Contextual representation to cope with the changes and keep the process adapted to the environment.

The scope of this study is to assess the process configuration, more specif-

ically the models generated by the BVCCoN approach. As object of study we selected the airport processes, a dynamic domain where changes in environment may impact the business processes' performance. The airport process domain is a well-known process that is already described in literature. Based on information from airport control agencies (de Wandeler 2008), as well as some projects that describe airports (Chatel et al. 2011, Ploesser, Peleg, Soffer, Rosemann & Recker 2009) we collected the data necessary to compose the simulation scenarios and the business process models. Describing simulation scenarios based on real data helps to obtain results closer to what would be expected in real situation.

The BVCCoN approach was designed to be used at run-time. It would be expected that the simulation assessment also consider this aspect. However, the simulations were performed with the definition of models and simulation scripts at design time. Figure 5.1 shows a schematic representation of simulation design. Observer that all models were designed before starting the simulation, including the BVCCoN models. That means, the changes in BVCCoN process were not performed during the simulation itself, but the simulation scripts were designed to mimic these changes. The definition at design time was necessary due to methodological issues and limitations in the simulation engine we could not design a dynamic simulation. First, the simulation design required the separation of the simulation script in order to support the parameters control. Moreover, the simulation engine could not support integration with BVCCoN to allow dynamic simulation. It was a decisive point that motivated the design of static simulation models.

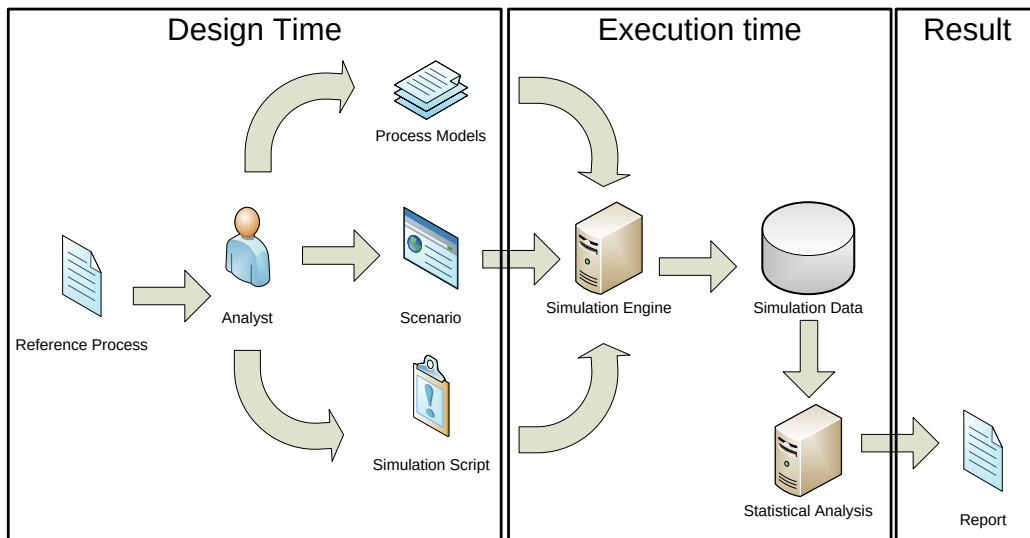


Figure 5.1: Schematic View of Simulation

In following sections we present the necessary background to understand this chapter as well as the design of the assessment and result analysis.

5.1 Background

In this section we present the template used to structure the assessment definition and report. This template organizes the assessment report avoid repetitions. The data description and explanations are located in specific sections. Moreover, we present background information about the statistical method used to answer the research questions.

5.1.1 Assessment template

Even though a simulation based study cannot be called an experiment in the strict sense of the term (Wernick & Hall 2007), the template to structure and report an experiment can also be used to a simulation study. Hence, we adopted a template proposed by Wohlin et al. (2000). In the sequel we detail the part of the template and how they are documented in this chapter.

Assessment Definition In the definition phase, the assessment foundation was defined. The purpose of this phase is to define the assessment goals. We used the structured framework (Wohlin et al. 2000) based on GQM which includes the object of study, purpose, quality focus, perspective and context.

Assessment Planning During the planning we defined the design of the assessment what includes definition of hypothesis and variables. The *Context Selection* section justifies the trade-off of using simulation as empirical tool. After that we detail the *Hypothesis Formulation* and *Variables* sections. The former state the hypothesis to be tested, and the latter, the variables that assessed to answer research questions. Last but not least, the *Assessment Design* section that describes the how we structure the assessment.

Assessment Operation The operation of the assessment requires several steps. The *Preparation* describes the steps performed before execute the assessment. It includes the definition of simulation scripts, as well as the settings of BVCCoN required to produce the process models such as the NFRs' priorities and the context values. In section *Simulation Models*, the process models are produced by BVCCoN following the assessment design. Then the simulations are performed in *Execution* section and the data is collected and packed in *Data Validation* section.

Results and Analysis In order to answer the research questions the data collected is analyzed and the hypothesis are tested. The *Descriptive Analysis* section describes and graphically presents interesting aspects of the data set. After that we can select a test to perform the *Hypothesis testing*.

Discussion Finally, we discuss the findings showed by the *Results and Analysis* section. In particular, we present in the *Validity of study* section we discuss threat to validity of the study and how we address them.

5.1.2 Mann-Whitney test

The Mann-Whitney test is a non-parametric alternative to the t-test (Wohlin et al. 2000). The parametric methods use the value of a parameter to compare the samples (e.g., execution time). In Mann-Whitney test, a non-parametric test, the comparison is made by the evaluation of rankings. In a same distribution sample, two populations (or classes) are ordered and compared. The position of the sample in the ranking as well as number of times a sample wins against the samples of other population are computed to calculate the U statistic. This statistic takes in account the ordinal position of a sample instead of the real parameter.

In parametric tests the statistical relevance is tested by the two-tailed p test. In order to test if a difference of U is statistically relevant it is necessary calculate a normal approximation called z . Thus, using U it is possible to calculate z that allows to test the null hypothesis.

In the next section we present the airport processes domain selected as case for the simulation description.

5.2 Airport Application Domain

As a motivating example, let us consider check-in related processes in the airport domain. Check-in is usually the first procedure for a passenger when arriving at an airport, as airline regulations require passengers to check-in by a certain time prior to the departure of a flight. This duration usually spans from 30 min to 4 h depending on the destination and the airline company. However, the airline check-in's main function is to accept luggage that is to go in the aircraft's cargo hold.

Several activities related to the airport check-in process are relevant and useful for the proper definition of business process models. For instance, passenger check-in policies and procedures regarding security, luggage handling, passenger handling and document validation. Check-in options and

procedures may vary according to the airline company, as some of them have certain restrictions that other carriers do not have in place. Occasionally the same company, at two separate airports, may have different check-in procedures.

This type of process takes place in a dynamic environment, which may be affected by several factors such as load of passengers, security policies, weather, and so on. In the above scenarios, it could be helpful to configure the process according to context changes, taking in consideration the quality preferences associated with the check-in process.

Note that we selected this domain, to assess our approach because it is dynamic. Thus it can be benefited from process configuration. Several airport's processes were already described in the literature (Chatel et al. 2011, de Wandeler 2008, Leggat et al. 2009, Maiden et al. 2004, Ploesser, Janiesch, Recker & Rosemann 2009, Ploesser, Peleg, Soffer, Rosemann & Recker 2009, Rosemann et al. 2008). Hence, we had plenty of material to research, which helped to identify variability. Moreover, some information used to configure the simulation environment is also available in reports from airport controlling agencies.

In the following sections we present a base check-in process model, and then proceed to present its variation points and variants. Next, we define which contexts are linked to the process variants. Last but not least, we present how these variants may impact the overall quality attributes of the process.

5.2.1 Reference Process

First we model a basic version of the check-in and boarding process that will be the basis for the approach. This process will be the reference model and should describe a valid business process. In Figure 5.2 there is a description of the check-in and boarding process. It does not contain detailed information such as exception handling and extra deviation in the main flow. Since the process model was used in a simulation engine, we also avoided the use of lanes and sub-processes, even though our approach is able to deal with this kind of details.

The check-in process starts when the passenger arrives at the airport and goes to the company counter. The airline staff starts by identifying the passenger and the flight. After that, the check-in is performed and a boarding card/pass is printed. If there is baggage to be sent to the aircraft cargo hold it is dropped here. Then the passenger can go to the security checkpoint where the security staff will check the passenger details. In this step the passenger may be asked to identify himself again, but we omitted this part

because the simulation engine we are using does not support sub-processes. Finally, the passenger is conducted to board at the gate.

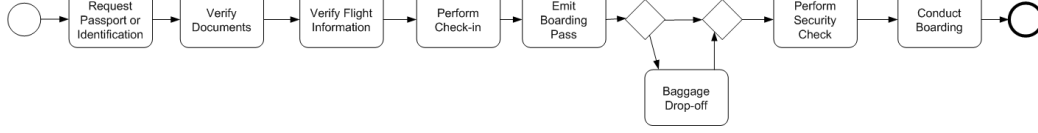


Figure 5.2: Figure Reference Process

5.2.2 Variability elicitation

The variability elicitation starts by analyzing the reference model. Based on the method proposed by (Liaskos et al. 2006) we derived a set of questions to help the identification of process variants (see Section 4.2). By applying these questions and examining the model elements we can identify some variations in the way the process is performed. For example, the Perform Check-in activity described in the reference process may hide details that could be important, such as who is responsible for performing the task (e.g., the Staff or Passenger), or the conditions associated with the task (e.g., the need to drop-off baggage) see Table 5.1.

This raw data will be analyzed and then represented as variants, variation point or as contextual information. It is up to the analyst to decide which data is actually relevant to the process in question. Based on the business analyst knowledge, the information is described as process fragments using BPMN. The process fragments are associated to variants and grouped in variation points, as it will be described in the following section.

5.2.3 Variability Description

The Variation Points indicate where the process can vary, we use *begins* and *ends* points to indicate where the variation point start and ends. In these points the idea is to indicate how the main flow can change. In the case of complex points with several in-coming and out-coming flows, it is possible to use a combination of variation points and dependencies (e.g., requires or excludes) to express them. The variation points may have a property (called operator: AND, OR, and XOR) that indicates how the variants are grouped. The AND operator is special role because in this case the business process fragments can be ordered as a sequence or in a parallel setting.

Figure 5.3 shows the graphical representation of the variation points in the reference process model. For example, the VP1 variation point begins in

Table 5.1: Question and answers for Perform Check-in task

Task	Question	Answer 1	Answer 2
Perform Check-in	Who will perform the task?	Check-in Staff	Passenger
	Who will be affected by the task?	Passenger	Passenger
	What are the objects consumed or produced by the task?	Paper Ticket	E-Ticket
	To what the degree the task will be performed?		
	How the action will be executed?	Check-in Staff confirm the check in	Passenger Perform on-line check in
	In what conditions the task will be performed?	When on-line check-in is not available OR the baggage needs to be checked	When on-line check-in is available AND There is no need to drop-off baggage
	Where the task will be performed?	Check-in Counter	internet
	When the task will be performed?	When the check-in for the flight is open	

the Verify Flight Information task and ends in the Emit Boarding Pass task, so everything between these tasks (including themselves) can be changed by the variants. There is also the case where just one task is a variation point – in this case the *begins/ends* points are linked to the same task – as we can see in VP2 and VP3.

The variants are associated to business process fragments. The use of process fragments allows the business analyst to describe coherent processes that will still be valid after the process change. However, this requires that it is necessary to assure that the granularity of the process parts are equivalent to the variation point, and that the *begins/ends* of both match.

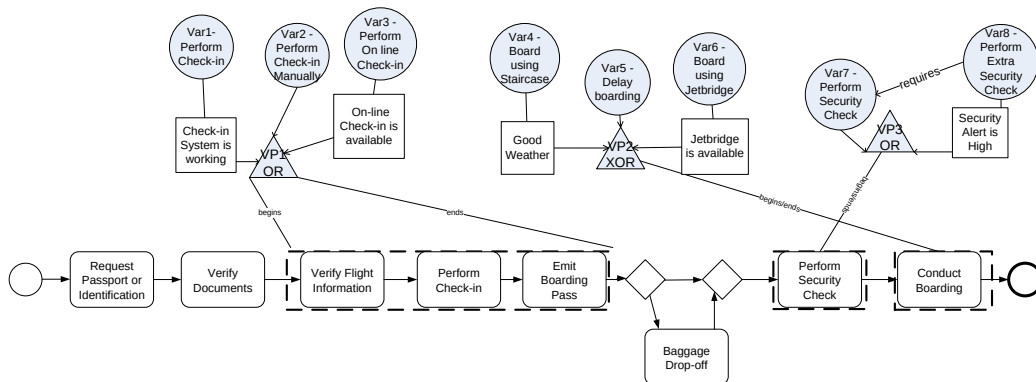


Figure 5.3: Variation Points in the reference process

Figure 5.4 shows the association of variants and business process fragments. See that the variants can be a single task - such as the Board using Staircase task (var4) - or be a set of process elements as the Delay boarding variant (var5). In some cases a variant may require the presence of other variants. For example, the Perform Extra Security Check variant (var8) requires the execution of the Perform Security Check variant (var7). In this case the opposite does not hold, since it is not necessary an extra security check when perform a simple one (standard).

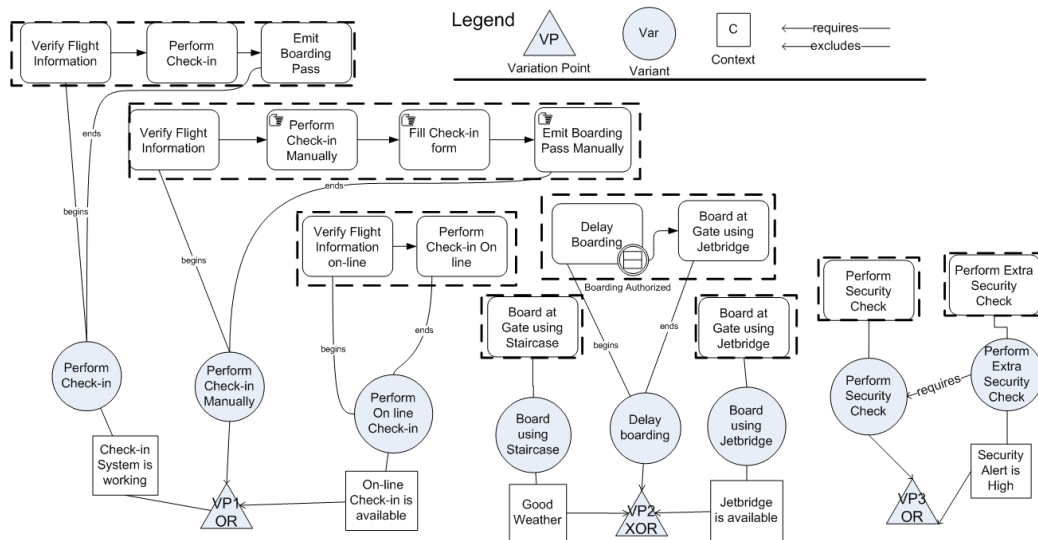


Figure 5.4: Variation Points and variants with business process parts

5.2.4 Contextual Analysis

After the description of variability, the next step is to associate contexts to process variants. The contexts describe states of the world that can affect the process. In some cases the process may be affected by several factors that can be originated inside the organization or in some other circumstances out of its control, such as the availability of resources or changes in weather conditions.

We represent the context information through logical expressions that can be evaluated through data monitoring. For example, the context data may be obtained by sensors, by consulting information services, by analyzing profiles or by users' inputs.

In the check-in example we identified the following relevant contexts:

Good Weather conditions The weather condition may affect the boarding process - if the weather is not favorable using a staircase to board may be not viable, and sometimes even the boarding itself could be delayed.

Internet and check-in kiosks are working some airline companies allow check-in through the Internet or by providing kiosks to the passenger. However, these resources must be available in order to be a valid alternative.

Jet Bridge is available Some large airports have jet bridges (Air Bridge, or just bridge) to allow the passengers to board at the gate without the need to go the aircraft from the runaways. They allow a faster boarding but are not available in all airports, as some small surface airports just allow boarding using staircases.

High alert security The airport controller or government agencies may issue a security alert that impacts the security checkpoints in airports. If this alert is in place the airport must change the way they check passengers before the flight.

On Line Check-In is Available In some exceptional cases the check-in system may become unavailable. The airline staff may switch to use manual check-in. Even though they are related, this is a different case from the Internet and check-in kiosks context. If the Internet services and kiosk are not working the passenger still will be able to perform check-in at airline counter, however if the check-in system fails neither will be valid.

5.2.5 NFR analysis

The last modeling step is the association of variants with quality attributes. We propose using NFRs to describe the quality attributes that may be relevant for the analysts. Catalogues may be built or reused to do this analysis. In our approach we use the qualitative analysis, where the analyst expresses the way he believes a variant may affect the process configuration in a controlled range: from more positive (++) to more negative (--). Based on this analysis and using prioritization it is possible to select a configuration of business process that is closer to the analyst expectations.

Figure 5.5 shows the contribution analysis for Security, Performance and Reliability NFRs. For instance, the Perform Security Check variant contributes positively to Access Control by reducing the access of people that should not enter to boarding area. However, this security check has a negative impact to Performance, as it reduces the speed of the process. The NFR graph may become complex and hard to read. Hence, we rewrote the contributions of Figure 5.5 into Table 5.2. The contributions create a sparse matrix where the white spaces will be substituted by Equals (=) labels for further computation effects.

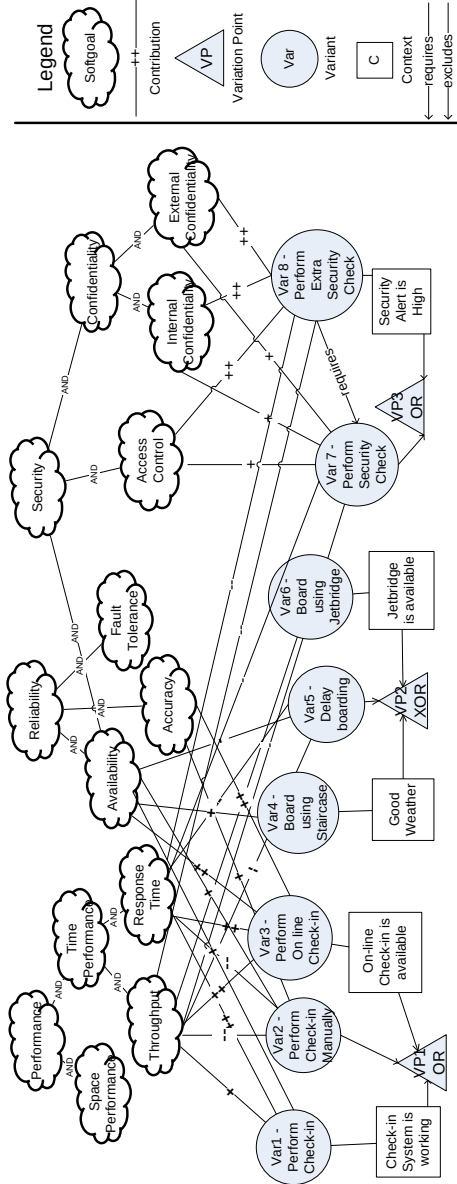


Figure 5.5: NFR and contribution analysis

Table 5.2: Contributions to NFRs

NFRs		var1	var2	var3	var4	var5	var6	var7	var8
Performance	Time Performance	+	-	++	+	--	++	-	--
	Response Time	++	--	++		--		-	--
Reliability	Availability	+	+	++	+	-			
	Accuracy	+	+	++					
Security	Confidentiality							+	++
	Internal Confidentiality							+	++
	External Confidentiality								++

Note that the result of this analysis is dependent on the analyst. However, it allows the evaluation of several quality attributes at once. Moreover, if the analysts believe that some NFRs that make sense for software systems do not make for a business process (such as recoverable) they can rebuild the NFRs catalogues and change them to fit their domain. Since quality attributes may exist in every knowledge area it is not hard to find information to update the catalogues.

Having illustrated our proposal, we now would like to be able to compare it against a standard process. In particular we would like to demonstrate that our approach generates more adaptable processes. Hence, in the next session we describe an empirical evaluation of it.

5.3 Assessment Definition

In this section we present a quantitative investigation performed in order to evaluate some characteristics of BVCCoN approach. Our goal is to evaluate the models produced by our approach in particular we would like to show that they are more adaptable to the environment, in terms of performance, than the standard process. Bellow we describe a structured description of the assessment objective:

Object of study. The object of study is the business process configurations generated using our automatic configuration approach. These models are going to be compared with respect to a basic process using the same scenarios.

Purpose. The purpose is to evaluate the business process configurations generated with our approach, verifying if they actually improve the process in different context settings.

Perspective. The perspective is the researcher's point of view.

Quality Focus. The focus is on the performance of a given process configuration running in a simulated environment. Two dimensions of performance are considered: the time required executing a process; and the resources required to execute a process.

Context. The context selected was the airport check-in and boarding process. In this context there are situations that may require changes in the business process to adapt to new scenarios.

5.4 Assessment Planning

In this section we detail the simulation study planning and describe the hypothesis and variables used during the study.

5.4.1 Context Selection

The context of the study is the execution of processes in an airport. It is conducted by a graduate student with focus on the processes dealing with check-in and boarding. The simulation of process is an empirical tool the provides several benefits (Wernick & Hall 2007):

- Accelerate generation of interesting new empirical evidence about efficiency and effectiveness of techniques and tools;
- Standardized representation and packaging of empirical evidence about local effectiveness and efficiency of techniques and tools in varying contexts, facilitating the systematic exploration of the impact on (global) performance at low cost;
- Improved knowledge transfer, education, and training through visualization of the impact of local effects on global performance.

Moreover, the simulation also allows combining several contexts and scenarios with low cost and risk when compared to an actual experimentation. We deal with events that may be out of our control (weather changes, demand peaks, etc.) during a real situation, using simulations they may reproduced in a controlled environment. Some activities that are required during a quantitative study such as control the variables interference and collect the resulting data are benefited by the use of simulation.

5.4.2 Hypothesis Formulation

In this quantitative study we focus on the evaluation of BVCCoN approach considering the models generated.

In some cases the models produced by BVCCoN must support the changes and maintain the basic level of performance.

In the sequel we state research questions and define the respective sets of null and alternative hypotheses:

RQ1: Considering the use of NFRs, do the models produced by our approach consume less resources and require less time to execute in comparison with a standard business process configuration?

RQ2: Considering the adaptation to a context change, do the models produced by our approach consume less resources and require less time to execute in comparison with a standard business process configuration?

From these research questions, we generated the following hypothesis:

Null Hypothesis $H_01.1$: The business process models produced by BVC-CoN approach, when considering the use of NFRs to configure the models, do not require less time when compared the standard process;

Null Hypothesis $H_01.2$: The business process models produced by BVC-CoN approach, when considering the use of NFRs to configure the models, do not require less resources when compared the standard process;

Null Hypothesis $H_02.1$: The business process models produced by BVC-CoN approach, when considering the adaptation to context changes, do not require less time when compared the standard process;

Null Hypothesis $H_02.2$: The business process models produced by BVC-CoN approach, when considering the adaptation to context changes, do not require less resources when compared the standard process.

If the null hypothesis can be rejected with relatively high confidence, it is then possible to formulate alternative hypotheses:

Alternative Hypothesis $H_a1.1$: The business process models produced by BVCCoN approach, when considering the use of NFRs to configure the models, consume less time when compared the standard process;

Alternative Hypothesis $H_a1.2$: The business process models produced by BVCCoN approach, when considering the use of NFRs to configure the models, consume less resources when compared the standard process;

Alternative Hypothesis $H_a2.1$: The business process models produced by BVCCoN approach, when considering the adaptation to context changes, consume less time when compared the standard process.

Alternative Hypothesis $H_a2.2$: The business process models produced by BVCCoN approach, when considering the adaptation to context changes, consume less resources when compared the standard process.

Having defined the hypothesis we need to identify dependent and independent variables.

5.4.3 Variables

Independent Variables. The independent variable of this study is the modeling method it can assume two possible values: **Standard** or **Adaptable**. **Standard** means a traditional BPMN model, without variability, while **Adaptable** is a process model generated with our approach, including variability, contextual and non-functional information.

Dependent Variable. The dependent variable is the performance measured using the execution time (in hours) and resources consumption (in cost). Resources consumption includes personnel costs, equipment costs, fares, and so on. Both dependent variables were calculated by the simulation engine, considering estimated time and resources required to perform each activity of the process. Note that the dependent variables are inversely proportional to the performance, i.e. the lower the value of execution time and of resources consumption, the better the performance. The measure variables were the average execution time spent in hours and the average resources cost expressed in dollars. Hence, the following dependent variables have been defined:

Execution Time The execution time is the average number of hours that a process instance needs to execute. It is calculated by the simulation engine considering the time required to execute the tasks of a process. The value is derived from: the execution time that describes the usual case, the estimated time that describe how much the execution may vary, and the maximum estimated time, which defines the maximum acceptable variation for the execution time. For instance, a *Perform On line Checking* task may be executed in 5 minutes with a variation of 5% in the estimated time and 20% in the maximum time. Hence, during the simulation the task may be executed from a minimum of 4.75 minutes to maximum of 6 minutes.

Resources Cost Another variable that may be evaluated is the cost of resources consumed during the process execution. The resource cost is the average value of a resource consumed during the process. It is calculated by the simulation engine considering the number of resources available and consumed as well as the cost of use. The cost of use may take in consideration the number of times a resource is used as well as its cost per unit of time, etc. Once the variables are defined we can move on to design the quantitative assessment.

5.4.4 Assessment Design

A basic requirement of an simulation based study is the ability to control the object of study and its parameters. Since our approach allows the modification of several parameters first we need to control the parameters that will be modified. In our case, we designed the simulation based study to assess the impact of NFRs' priority change as well as the impact of contextual changes. Hereafter, when changes in NFRs are mentioned we mean changes in the NFRs' priority. Likewise, the contextual changes means changes in the values of the context variables.

We combined the principles of blocking and balancing in the design of our study. The blocking is a design principle that is required in the definition of assessment. It is applied when we have a factor that probably has an effect on the response, but we are not interested in that effect (Wohlin et al. 2000). When the effect is known and controllable the blocking technique may be used. Blocking is used to systematically eliminate the undesired effect in the comparison among treatments.

The balancing technique is used to assure that each treatment to be studied have the same number of subjects. Balancing is desirable because it both simplifies and strengthens the statistical analysis of the data.

The two scenarios are considered:

- S0.** Vary NFRs' priority to identify their impact on the new processes generated by the BVCCoN approach.
- S1.** Vary contexts to identify their impact on the new processes generated by the BVCCoN approach.

Our design setting blocks the variation of NFRs and contexts in order to isolate the interference among them during configuration.

In S0 we vary the NFRs' priorities and fix the contexts. This generates the processes that reflect just the changes in NFRs. On the other hand, in scenario S1, we vary the contexts to generate new processes while the NFRs' priorities are fixed. Thus, all processes generated in S1 have the same NFRs' priorities. With that we can compare the results of changes without the interference of other factors in the simulation.

Another important aspect of the quantitative study is the way that we plan to balance the simulations. In order to obtain a balanced simulation, we repeat each simulation for each configuration of contexts and NFR for both standard and adaptable processes. When a context is set as active some variables of simulation change such as the availability of a resource. The simulation setting to reflect this change is then used during the simulation

of the standard process and an adaptable process generated for that context change. Note that the number of simulations in the study is balanced according the situation.

The S0 and S1 scenarios were proposed to address the research questions *RQ1* and *RQ2* respectively. This separation among NFRs and contexts changes in different scenarios simplifies the design and execution of our assessment. First, the conditions present in the simulation may be repeated within each scenario which assures that the process simulation will be executed in equal bases. A context change may affects the presence of a resource in scenario S1, that in its turn may affects tasks that were prioritized by a NFR. However, it is not a desirable interference because we cannot predict if the change was caused by a NFRs' priority change or a context change.

A standard check-in and boarding process described in the literature (Ploesser, Peleg, Soffer, Rosemann & Recker 2009) was considered as the standard business process model (see Figure 5.2). We select this process because it was already described in literature and validated against in a real situation. Afterwards, we used our approach to design a new business process models based on the change in the S1 scenario. For each, scenario we simulated the case with and without contextual change. For instance, the *Perform on-line check-in* and *Use jet-bridges to board* variants can improve the performance of the process by reducing the waiting time. However, this type of alternative can increase the costs or may be unavailable at the moment when the process needs to be executed.

The BVCCoN approach was designed to be used at run-time. It would be expected that the simulation assessment also consider this aspect. However, the simulations were performed with the definition of models and simulation scripts at design time. All models were designed before starting the simulation, including the BVCCoN models. That means, the changes in BVCCoN process were not performed during the simulation itself, but the simulation scripts were designed to mimic these changes. The definition at design time was necessary due to methodological issues and limitations in the simulation engine we could not design a dynamic simulation. First, the simulation design required the separation of the simulation script in order to support the parameters control. Moreover, the simulation engine could not support integration with BVCCoN to allow dynamic simulation. It was a decisive point that motivated the design of static simulation models.

In order to describe the impact of the context in the process we based our estimate in the data related to the air traffic delay recently experienced in European airports (de Wandeler 2008). The simulation data requires the estimate of the tasks duration, the resources costs, and the number of involved personal. For example, the collected information could help to estimate how

much time is necessary to perform the *Board at Gate* task. As consequence of being based in real information the simulation scenarios are closer to the real situation.

In S0 we described a simulation setting with fixed resources and execution conditions when used the BVCCoN approach to generate several process instances. The adaptable processes were simulated using the same simulation setting as the standard process. It was repeated until we obtained the same number of executions for each type of process (i.e., adaptable and standard). After that a comparison was performed using the results.

The S1 scenario received a different treatment, since the context changes necessarily modify the simulation conditions. Several conditions were defined and executed for the standard process and for an adapted process designed for that conditions.

In summary, for S0 we have different processes for the same simulation settings, and in S1 we have a process for each different simulation settings.

For S0 we may use the t-test and Mann-Whitney test (Wohlin et al. 2000) to verify if there is difference between the standard and adaptable processes. The S1 scenario may be evaluated using the same tests since we are comparing the standard and adaptable process variables. The selection of statistical test depends on the resulting data since t-test requires a normal distribution of data.

Once the assessment was designed, the next step is performing it. The following section presents the preparation and models used in these study.

5.5 Assessment Operation

5.5.1 Preparation

The quantitative study was performed using the simulation feature of the Bonita Open Solution 5.6 business process platform¹. This environment allows the representation of the process data, simulation of the environment resources available for the execution of the activities as well as the configuration of time taken. Hence, we were able to control the variables of the assessment, allowing the reproduction of the study by independent teams.

Using the BVCCoN approach we designed several processes for S0 and S1 scenarios. In S0 scenario the variation of NFRs was performed using the for Performance, Security and Reliability priorities of BVCCoN model from table 5.2. The priorities of these top level NFRs were derived from priorities given to leaf NFRs such as Throughput or Availability. After computation using

¹www.bonitasoft.com

the selection algorithms (i.e., the AHP) we obtained the variants preferences shown in table 5.3. The values are used to select the most preferred values, that means the higher the better.

Table 5.3: Computation of NFRs' priority over Variants

Task	Performance process	Reliability process	Security process
Perform Check-in	0.16850276	0.1561380421	0.1056760263
Perform Check-in Manually	0.1059536981	0.1209307344	0.0923948759
Perform Online Check-in	0.2677317084	0.2699253717	0.1684273085
Boarding Using Staircase	0.1320638366	0.1288088495	0.0941969976
Delay boarding	0.0486993775	0.0530210609	0.0517413584
Boarding using Jet bridge	0.1364798598	0.1234999692	0.0880050471
Perform Security Check	0.1405687596	0.1476759723	0.1525170352
Perform Extra Security Check	0.00109758	0.00107586	0.2470413511

Note that columns represent the process derived for a given NFR priority while the lines capture the variants and their preference for each NFR priority. These values are numerical indicators of preference, the higher they are the more important (priority) they become. Observe that the computation of the contribution and priorities results in general preference of variants. Hence, we defined four processes to be simulated in scenario S0, three addressing the NFRs listed in table 5.3 plus the standard process.

Since, in the S1 scenario changes the NFRs' priority are not allowed, the value of priority was not used in the definition of new processes. Instead, we defined the following settings for changes in contexts:

Without Internet The tasks that requires Internet resource to be performed such as *Perform Check-in* and *Verify Flight information* are affected by the lack of this resource. The simulation of this context was designed using the configuration capability of the simulation engine that allows the definition of when and how much of the resource is available. In this case the amount of time the Internet is not available during simulation.

Without Online Checking This context change was designed supposing that only the check-in at the airline counter was available during a period in the simulation.

Small Airport This context change means that some resources (such as *Jet bridges*) that are commonly available in larger airports are not available. The simulation of this case is performed removing the *Jet bridge* resource from the simulation engine. This results in processes where the variants that depends on *Jet bridge* resource are not able to run during some time.

Security issue In this case a security alert was announced, the simulation of this context change was designed using the conditional execution capability of the simulation engine. Rules with the description of conditions activate resources or generate delays during the simulation to reproduce the execution of this task. For instance, when the security alert is issued the processes that are not prepared to perform an extra security check will receive a delay according the data from de Wandeler (2008) for each process executing.

Given these contexts we designed a process for each of them using the BVCCoN approach as well as a standard process.

5.5.2 Simulation Models

The simulation scenarios are based on how the resources available are described for the simulations. The variables selected are related to resources and to the estimation of each activity execution. Based on the information of de Wandeler (2008), Leggat et al. (2009) we estimate the values of time in execution of the tasks for each process. Table 5.5 describes the execution time, estimated time, and maximum time. The execution time is the duration of the task during the simulation. The estimated time and maximum time are the margins of the simulation execution uncertainty. They are expressed in terms of the percentage of variation a task is expected to produce and the maximum acceptable. Other resources used during the definition of simulation scripts are available in the appendices, including the process models (see Appendix C) and the relationships among contexts, tasks and NFRs (see Appendix D).

Another simulation aspect designed was the resources available during the simulation. Recall that the resources may be consumable or renewable. The consumable resources exhaust after consumption and may be not used again and they have a fixed cost per use. On the other hand, the renewable resources may be used again and again and has a cost by time of use. The cost of resources was defined according with review of reference material. The cost of a staff personnel, such as check-in staff and security staff, was defined

Table 5.4: Simulated time of tasks

Task	Execution time (minutes)	Estimated time (%)	Maximum Time (%)
Perform Check-in	5	15	30
Perform Check-in Manually	5	15	40
Perform Online Check-in	3	10	20
Boarding Using Staircase	15	30	100
Delay boarding	60	25	50
Boarding using Jet bridge	10	15	50
Perform Security Check	5	30	75
Perform Extra Security Check	10	50	100
Verify Flight information	3	10	20
Request Visa/Documents	5	50	100
Verify Documents	5	50	100
Baggage Drop-off	5	50	100
Emit boarding pass	2	15	50
Emit boarding pass Manually	5	30	50
Verify Flight information On line	1	10	20
Fill check-in form	5	30	50

based on average annual salary of these employees. The cost of physical resources such as the x-ray and Internet service was defined based on the average operational cost.

5.5.3 Execution

The execution was performed using the Bonita Software business process simulation engine. During the execution two separated approaches were necessary. In the S0 scenario, where the process were adapted to reflect NFRs prioritization, the four processes were simulated using the same simulation settings (resources and time tables). The three processes that were adaptable dependent variable were simulated 30 times each. The process that uses the standard variable was simulated 90 times. Hence, we hope to have a balanced set of results. Since the simulation based studies may test exhaustively all

Table 5.5: Simulated cost of resources

Resource	Maximum	Availability	Cost (\$)
Jet bridge	2	by instance	1.00
Internet service	1	constant	0
Security Staff	2	by instance	15 per hour
X-Ray	1	constant	80000 per year
Check-in staff	2	by instance	10 per hour
metal detector	1	constant	5 per hour
Boarding Staff	5	by instance	10 per hour
StairCase	2	by instance	0.70

scenarios, we use 30 times as the default number of runs for each simulation. This is necessary in order to reduce sampling bias allowing balanced comparison among simulations.

In the S1 scenario, the simulation setting varies according to the context configuration. Thus, to obtain a balanced set of results considering these variations we executed the simulation in a paired arrange. The simulation setting was prepared for the execution and an adapted process and a standard process were executed for this configuration. Then a new simulation setting is prepared for another pair. After the execution we obtain 120 results for the adaptable processes and 120 for the standard process.

The Bonita Software Simulation engine produces simulation reports in PDF files, each simulation corresponds to a report file. The simulation reports were manually transcribed into spreadsheets to further analysis.

In general, the design, execution of the simulation as well as the data collection required almost four months of work. We estimate in 550 man-hours spent in these activities.

5.5.4 Data Validation

Data was collected from the simulation reports and tabled to allow analysis. The simulations that crashed were replicated to fill the missing data as well to maintain a balanced number of results.

In some cases the simulation presented extreme variations in the results. During seminar with senior researchers we were reminded that these variation are also important and need to be evaluated during the analysis of results step. All data was considered during analysis reported in this chapter.

5.6 Results and Analysis

5.6.1 Descriptive Analysis

As a first step in analyzing data, we use descriptive statistics to visualize the data collected. We employ a statistical tool to help to analyze and test of hypothesis - the SOFA Statistics ². This tool tests the data for normality and applies statistical test to answer the research questions.

Figure 5.6 presents the histograms of collected data from both simulation scenarios considering the adaptable and standard processes. First, in Figure 5.6(a) and 5.6(b) we see the distribution of the execution time for the scenario S0 considering the standard and adaptable processes. The graphics do not present a clear normal distribution of the results of both variables of this scenario. Indeed, none of the data set passed the test of normality using the SOFA Statistics tool. However, it does not mean that they cannot be analyzed, since real world data may also fail the normality test (Wohlin et al. 2000). Likewise, the data in the chart of Figures 5.6(c) and 5.6(d) also do not present a normal distribution.

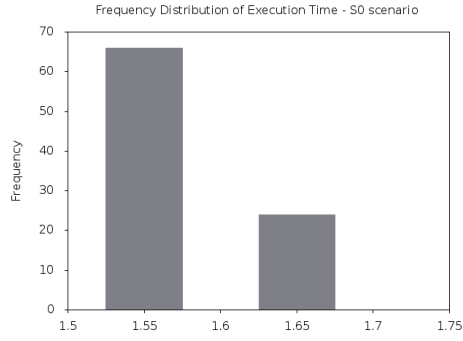
When considering the resource cost, it is possible to identify that the results are very behaved in some conditions. Observe in Figure 5.6(c) that about 30 samples are in the same group of values between 4.4 and 4.5. While there are about 60 samples range between 4.5 and 4.6.

When considering the results of resource cost for adaptable process in Figure 5.6(d), the same pattern is detected, there are two ranges of values: between 3.5 and 4, and between 4 and 5. However, the rest of samples does not follow this pattern.

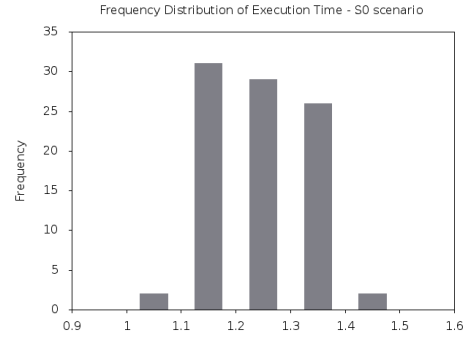
Figure 5.7 presents the data distribution for the S1 scenario where the context triggers changes in environment variables. According to the data distribution shown in the histograms, it is possible to observe that the data does not pass the normality test. On the other hand, if we consider the execution time of the adaptable processes (see Figure 5.7(b)), it is possible to identify a distribution that is close to the normal curve. In order to improve the presentation, we divided the distribution in dependent variables like in Figure 5.7. However, during the execution of statistical evaluation all the data shall be considered. Hence, we conclude that the data do not pass in normality test.

After evaluating the data we identified that it is not possible to apply statistical hypothesis tests that require normality as a pre-requisite, such as t-test. Thus, we selected the Mann-Whitney U test (Wohlin et al. 2000)

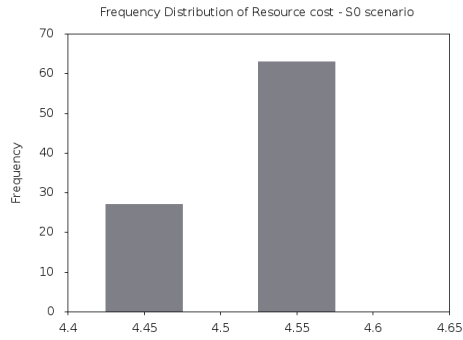
²<http://www.sofastatistics.com>



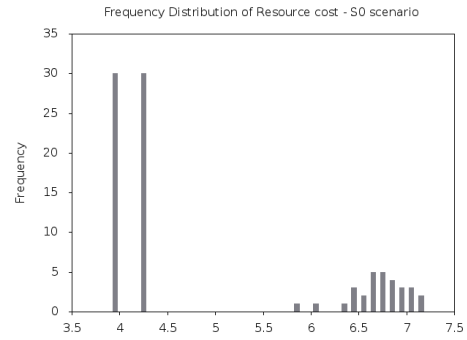
(a) Histogram of execution time, standard process



(b) Histogram of execution time, adaptable processes



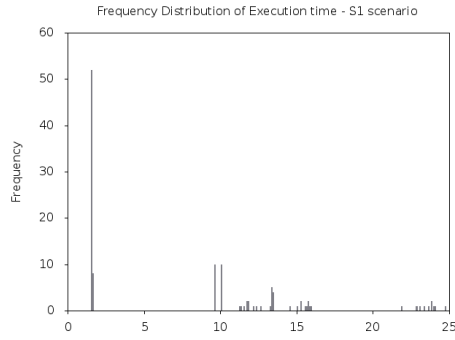
(c) Histogram of resource cost, standard process



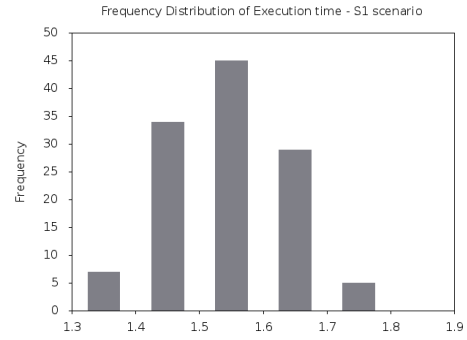
(d) Histogram of resource cost, adaptable processes

Figure 5.6: Data distribution for S0 scenario with variation of NFR priority

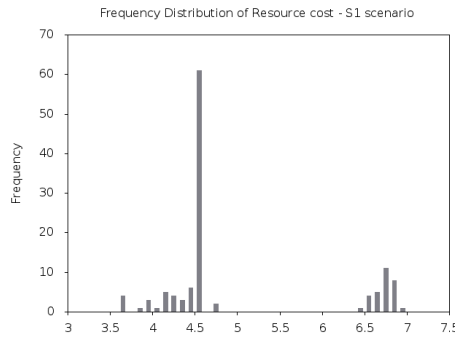
to evaluate the hypothesis, which is an alternative if the assumptions made by the t-test seem uncertain (i.e., normality). In next section we present the hypothesis testing to answer the research questions of this quantitative study.



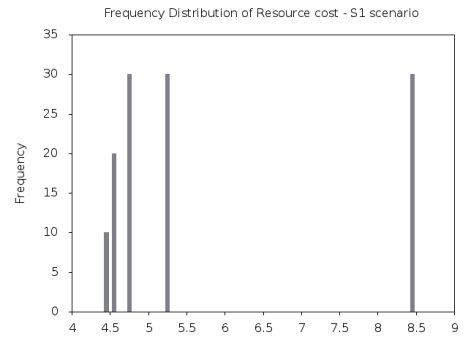
(a) Histogram of execution time, stan-



(b) Histogram of execution time, adapt-
able processes



(c) Histogram of resource cost, standard
process



(d) Histogram of resource cost, adapt-
able processes

Figure 5.7: Data distribution for S1 scenario with variation of contextual information

5.6.2 Hypothesis Testing

The Mann-Whitney U test is a non-parametric statistical hypothesis test for assessing whether one of two samples of independent observations tends to have larger values than the other (Wohlin et al. 2000). The test consists in comparing values in scales that present order such as numerical or ordinal sets. During the test the values are substituted, ordered and ranked. The evaluation of the rankings results in a statistics that indicates how many times a specific class has larger values the other. The test also provides significance indicators that allows the analysis if the difference was significant or not.

As mentioned before, we used the SOFA Statistical tool to perform the hypothesis testing. The tool reported results of test presenting the p value, U statistic and the statistics of the data presented in a table. The p value indicates if the test was statistically significant. Values bellow 0.01 are a good indicator of statistical significance. When the p value is below 0.001 the report does not present the actual value, just the declaration that the difference is relevant. In general the U statistics uses a table with critical values to identify if it is statistically relevant. However, when the number of samples is higher than 20 the test uses an approximation called z that allows calculate the p value. Thus, the U statistics may help to understand the degree of significance.

S0 scenario - Variation of NFRs

We compare the standard process against the processes generated by the BVCCoN approach considering the variation of NFRs. We have two null hypothesis $H_{01.1}$ and $H_{01.2}$. They refer to execution time and resource cost respectively.

In order to test the first hypothesis we applied the Mann-Whitney U test to the data sample generated based on the S0 scenario. The sample classes were divided in two groups of 90 items each. U is based on the results of matches between the *Adaptable* and *Standard* groups. In each match, the winner is the one with the highest *Execution time* (in a draw, each group gets half a point which is why U can sometimes end in .5). The resulting U value was $U = 0.0$, which means that all elements of *Adaptable* class has a smaller value than elements of *Standard* class. The further the number is away from an even result, that is, half the number of possible matches (i.e. half of $90 \times 90 = 4050.0$) the more unlikely the difference is by chance alone and the more statistically significant it is.

After calculating the U value the test derived $z = 11.595$ value. We also

performed a p-test resulting in Two-tailed p value is less than 0.001. If p is small, e.g. less than 0.01, we can assume the result is statistically significant, i.e. there is a difference. With this **result we can reject the Null hypothesis** $H_01.1$ with high degree of confidence and assume the alternative hypothesis $H_a1.1$ is true: The business process models produced by BVC-CoN approach, when considering the **use of NFRs** to configure the models, **consume less time** when compared the standard process.

Table 5.6: Statistics of execution time variable

Group	N	Median	Avg Rank	Min	Max
Adaptable	90	1.268	45.5	1.074	1.436
Standard	90	1.587	135.5	1.512	1.6

The table 5.6 presents the statistics produced by the test. Analyzing the minimum and maximum values for execution time we can see that the minimum value of standard process is larger than the maximum value of the adaptable class. There were no intersection among the values of the two classes. It explains why the U test scored too small value ($U = 0.0$), and the z value was too high ($z = 11.595$).

In the hypothesis testing of $H_01.2$ the result was $U = 2.700.00$. Recall that in each match, the winner is the one with the highest *Resource cost* (in a draw, each group gets half a point which is why U can sometimes end in .5). The further the number is away from an even result i.e. half the number of possible matches (i.e. half of 90×90 i.e. 4050.0) the more unlikely the difference is by chance alone and the more statistically significant it is.

Based on the U value $z = 3.877$ was derived and obtained a Two-tailed p value is less than 0.001. Hence, it is possible to **reject the Null hypothesis** $H_01.1$ with high degree of confidence. Therefore, the alternative hypothesis $H_a1.2$ is true: The business process models produced by BVCCoN approach, when considering the **use of NFRs** to configure the models, **consume less resource** when compared the standard process.

Table 5.7: Statistics of resource cost variable

Group	N	Median	Avg Rank	Min	Max
Adaptable	90	4.242	75.5	3.928	7.158
Standard	90	4.516	105.5	4.494	4.544

The data in table 5.7 presents a different behavior when compared with the execution time. The values of the standard process were within the range of the adaptable processes, which present a lower and higher value for cost.

However, the median shows that the distribution of values of the adaptable process was closer to the minimum value. This makes the adaptable class gain more matches against the standard process resulting in a statistically significant result.

In summary, considering the variation of NFRs' priorities the process produced by BVCCoN approach reduced the execution time and resources cost when compared against the standard process.

S1 scenario - Variation of Contexts

In S1 scenario we compare the standard process against the process generated by BVCCoN approach considering the **variation of contexts**. In this scenario 120 samples of each class were used in the comparison. Again, we employ the Mann-Whitney U test to compare the values of execution time and cost for both classes.

Considering the null hypothesis $H_02.1$ (we evaluated the execution time variable), we obtained $U = 3165.0$. Recall that the further the number is away from an even result (half the number of possible matches i.e. half of 120×120 , or 7200.0) the more unlikely the difference is by chance alone and the more statistically significant it is. Based on this value, the $z = 7.505$ was derived and we obtained Two-tailed p value is less than 0.001. With that value it is possible to **reject the null hypothesis** and assume the alternative hypothesis $H_a2.1$. In other words the business process models produced by BVCCoN approach, when considering the **adaptation to context changes**, **consume less time** when compared the standard process.

Table 5.8: Statistics of execution time variable

Group	N	Median	Avg Rank	Min	Max
Adaptable	120	1.535	86.875	1.366	1.711
Standard	120	5.611	154.125	1.512	24.738

After evaluating the data from 5.8, it is clear that the values of execution time for the adaptable group are within a closer range. The standard processes, on the other hand, has a greater range. Indeed, the median value shows execution time for standard process is far from maximum value for adaptable process. This means that at least half of the samples of standard process present a higher execution time the adaptable processes.

The last hypothesis test is the null hypothesis $H_02.2$. This hypothesis states that the business process models produced by BVCCoN approach, when considering the **adaptation to context changes**, do not require **less resources** when compared to the standard process.

When we apply the Mann-Whitney U Test, to compare the *Resource cost* for *Adaptable* vs *Standard*, we conclude that there is a difference with $U = 4379.0$ and $z = 5.248$. The two-tailed p value is less than 0.001, which indicates that the difference is also statistically relevant. However, in this case the difference is in favor of the standard process. As a result we cannot reject the null hypothesis H_0 2.2.

In table 5.9 we can observe that there is an intersection between the values of the adaptable and standard processes. However, the adaptable process present a higher maximum value and median what means that in general the values for the adaptable process are higher. In other words, the adaptable process produced by BVCCoN approach consume more resources than a standard process.

Table 5.9: Statistics of resource cost variable

Group	N	Median	Avg Rank	Min	Max
Adaptable	120	5.002	144.008	4.490	8.469
Standard	120	4.516	96.992	3.625	6.9145

5.7 Discussion

We began this chapter illustrating how our approach could be used in a real setting namely to describe check-in related process in the airport domain.

We relied on simulation, an empirical tool to present a preliminary assessment of our approach. We came to a conclusion that there is a relationship between how the process model behaves to adapt to the changes in the NFRs and Contextual information, to its execution time and resources cost.

According with the hypothesis test we were able to answer affirmatively that if: considering the use of NFRs, the models produced by our approach consumes less resources and require less time to execute in comparison with a standard business process configuration (*RQ1*). The hypothesis testing showed that our approach when considering the NFR changes indeed produced processes that consumed less resources and required less execution time to be performed.

Recall that the BVCCoN configuration process considers the preferences of stakeholders over variants. We showed that when the NFRs had been considered our approach generated appropriate variants that required less resources. Thus, a process configured by BVCCoN process selected variants that could reduce the resources consumption. In our example, the Performance NFR was used for the selection of variants. Even if the Performance

NFR was not defined as a priority NFR, the contribution for Performance still are taken in account during the selection. It happens due to the multi-criteria decision algorithm, which always consider all NFRs in some degree.

However, if we consider the results of hypothesis testing we are not able to provide a conclusive positive answer to the research question *RQ2*. This is because we have distinct answers to the execution time and cost analysis. When we consider the execution time the answer could be yes. However, when we consider the resource cost the BVCCoN approach does not produce a process that consumes less.

Evaluating the data we identify the following relevant points:

- Some simulations did not finished within the time limit of scenario S1. This happened during the execution of the simulations for the standard process, which is not able to adapt to some situations. The result is that in some simulations it did not consume all the resources as expected. The process was halted;
- The adaptation may require the addition of new tasks in the process as well as can increment the resource consumption. For example, when a security alert context is active the tasks referent to an extra security check may require more resources (e.g., security staff, x-ray or metal detectors).

Note that in the real world, when a check-in/boarding process is delayed beyond a certain limit (Leggat et al. 2009), the air line should cancel the flight and penalties are incurred. This aspect is not reflected in the simulation since it is external to the process. However, if we consider that even with a higher cost, the BVCCoN approach could produce processes that are able to finish in time, reducing the chances to get penalties. Moreover, small gains can add up to enormous benefits since this is a process need to execute by thousands of passengers that need to check-in.

As presented in Section 2.3.4, the use of AHP has a limitation about how to explain the results obtained by the method. Even though that the results are intuitively close to the preference (Liaskos et al. 2012), the AHP method still is a black box. This limitation is shared with data mining methods (Cortez & Embrechts 2011) that used numerical computations such as artificial neural networks or support vector machines. When we use the matrices calculations to compute the results, the direct link between the input (a possible preference setting) and the output (the selected configuration) is lost. Hence, we believe that the statistical evaluation presented in this chapter shows that our approach may be useful in some circumstances, even if we are not able to explain the results in details.

Changing the simulation time in order to wait all simulations to finish is not a viable strategy. First, we cannot ensure that all simulations will end. If the event that interrupted the flow is not reversible, the simulation may run forever. Moreover, in order to implement this strategy we should redo all simulation, which is not in question due time limitation to finish this thesis work. Thus, we believe that using a fixed simulation time was a valid strategy to obtain results for the assessment.

5.7.1 Validity of study

Among the threats to the validity of our study we identified the following items: the use of simulation as empirical tool simplifies some possible threats related with human subjects such as rivalry, motivation of teams, interaction among participants, and so on. Moreover, some threats may be present in the study. For example, the reliability of measures. Our measures were compiled from the simulation reports generated by the simulator.

There are also the design threats, such as the interaction among treatments and the mono-method bias. The former is caused if the subject is involved in more than one study, treatments from different studies may interact. The latter happens when using a single type of measures or observations involves a risk that if the measure or observation gives a measurement bias, then the study will be misleading. In order to reduce these threats we combine the treatment with different settings including the use of different NFRs such as Performance, Reliability, and Security as parameters for the process configuration, and repeated the simulation with and without contextual changes. The separation in two scenarios with different treatments, helped us to reduce this threat. Blocking the interaction among NFRs and context changes allowed us to draw conclusions about the influence of this factor in the final result.

Among the conclusion validity threats we identified that the low statistical power could be a problem, since non-parametric hypothesis testing had less valid statistics to be evaluated. The choice of a non-parametric test was driven by the need to adapt to the data distribution. Recall that during the descriptive statistical analysis we were not able to verify the normality assumption in the data sets. The non-parametric tests are less sensible to distributions that do not pass the normality test. Thus, we avoided the threat to violate the assumptions of a statistical test.

Other conclusion validity threat to fishing for a result, i.e. where the researchers induce the result to prove their hypothesis. Note that we evaluate the data and drew conclusions based on the results, considering even negative results as valid results. Even the hypothesis that was not rejected showed

relevant information to our study.

5.8 Summary

In this chapter we presented a quantitative study to evaluate the BVCCoN. It was based on the use of simulations, because being an empirical tool it allows the comparison among several scenarios of execution, related to lower risks and costs. With simulations we can test several context changes without wait for it to happen. The controlled environment also avoid risks in critical situations that may lead to disasters or economical loss.

The airport processes example was selected due to its characteristics. It is a domain where the processes are sensible to environmental changes. Moreover, it has been extensively studied in the literature (Ploesser, Janiesch, Recker & Rosemann 2009). It also has resources that allow the proper collection of process indicators (Chatel et al. 2011, Leggat et al. 2009). Hence, we can reproduce situations that are closer to real world.

Even though simulation based studies allow evaluation of results without hypothesis testing (Wernick & Hall 2007), we prefer use hypothesis testing to answer our research questions with more confidence. Confronting the data to demonstrate that there is statistical relevance in the results strengthens the conclusions.

Finally, the answer to our research questions indicated that our approach is relevant and may contribute to a better use of business processes. Even the negative results did not invalidate our approach, a careful analysis indicate that there are side benefits that may be highlighted in these situations.

Chapter 6

Conclusions

This thesis has investigated the dynamic configuration of process models as well as proposed an approach to configure process models based on Non-Functional Requirements and Contextual information. Business Process models allow the representation of the activities and resources of an organization, being essential part of the Business Process Management. When employed in dynamic environment it requires flexibility in process (Bentellis & Boufaïda 2008). Approaches with *a priori* implementation for process flexibility, that use the selection as main procedure, have limitations when applied during run-time. Among the limitations that we addressed there is the lack of high level criteria to configure business process and associated information that drive the configuration of the process model.

In Chapter 5 we presented a simulation based assessment to evaluate our approach in a controlled environment. The results indicated that our approach may improve the execution of process in terms of execution time and resource consumption. This was due to the use of NFRs as configuration criteria and context models as triggers to perform changes in models. Returning to the hypothesis/question posed at the beginning of this study, it is now possible to state that the use of NFRs and contexts indeed helps the configuration of business process.

6.1 Contributions

Following we highlight the main contributions of this thesis:

- The definition of an integrated metamodel to support configuration of business process models. The metamodel, Chapter 3, combines the concepts of Business Process model, context-awareness, and Non-Functional Requirements in the same metamodel. With that we pro-

vide a new form to represent these concepts, as well as use these new tools for process configuration.

- The definition of an algorithm for configuration of business process models that takes in account the preferences of stakeholders and contextual information. The algorithm is presented in Figure 4.11, Chapter 4. This algorithm integrates the selection of variants with the AHP method offering support for multi-criteria analysis of variants.

We now discuss some aspects of our approach presented in this thesis.

6.2 Discussion of Decisions and Findings

In BVCCoN approach, we chose to separate variability modeling from the business process model. It is less intuitive and requires more effort to describe variants, variation points and keep consistency with other models. However, several advantages may be identified such as no need to extend the business process modeling language and modeling tools. Hence, the input and output models may be exchanged and manipulated by several available modeling tools the only constraint is that they should also be based on the same metamodel used by our approach. The separation of variability description from the business process model also simplifies the maintainability of the model (i.e., inclusion, exclusion and modification) and the linking with other models (e.g., context analysis, and NFRs). Hence, the other analysis steps that require the manipulation of variability can be performed because they do not interfere with the variability model. Another interesting advantage is the scalability of the approach. If the variability information is embedded in the business process model, it would make the process model unusable if the number of variants grew too much.

Another critical decision related to the variability representation is the way to represent variation point and variants. We opted to have our own description of variability instead of using the common Feature Model (Czarnecki et al. 2005). However, this decision may impact the acceptance of BVCCoN because of the resistance of SPL community to use models other than those based on FD. We took this decision in order to avoid conceptual conflicts with other models. Observe that is a common practice in feature models to represent quality attributes as features (e.g., a security feature) grouping all variants that impacts them in the same node. Since we are interested in representing NFRs as configuration criteria, it was necessary avoid conceptual conflicts and represent the quality attributes in another single place (model). With the NFRs model we can represent quality attributes

that are transversal to all variants instead of being represented by just a group.

The separation of variability description also impacts the way we represent configuration knowledge. We selected NFRs as representation of preferences over variants, and contextual information to trigger changes in the model. Combined they will represent part of configuration knowledge necessary to create new process models. Of course this is less intuitive than using the representations of configuration knowledge present in SPLs (Bachmann & Clements 2005). However, BVCCoN compensates it by providing a rich representation of configuration criteria which allows the use of multi-criteria analysis algorithms.

The adoption of NFRs as configuration criteria is one of the key points in the solution proposed by BVCCoN. With some few exception (e.g., Lapouchnian et al. (2007)) the use of NFRs has been focused on the design phase (Xavier et al. 2010). NFRs describe preferences of stakeholders over variants allowing the selection of the preferred variants in the situation. However, some drawbacks emerge such as the extra design effort that will be required. Moreover, the contribution analysis requires a large computational effort to analyze the preferences, as the model grows the number of matches is multiplied. It is worth mentioning that in BVCCoN we are not dealing with the standard satisfaction notion of NFRs, as proposed by Chung et al. (2000), and also used to some extent by Lapouchnian et al. (2007). Since we are dealing with a mutable set of variants, the analysis of NFR satisfaction may not give proper results. For instance, let's consider that a variant (essential to satisfy a NFR) is not valid at a moment due to a context change; in the notion proposed by Chung et al. (2000) the NFR would not be satisfied. In order to address this limitation we applied the AHP (Saaty 2008) as part of the selection mechanism. Since AHP is a robust multi-criteria decision analysis method, it can deal with the inconsistencies and still produce a reasonable result.

Another relevant drawback is that the NFRs contribution analysis is basically a qualitative method that encodes preference in terms of satisfaction degrees. It reduces the possibility to use metrics as parameter to the configuration. Even though measures such as the Key Performance Indicators (KPI) are a common way to represent the critical success factors in business processes (Liaskos et al. 2007). Our approach does not address their use. Although, the adoption of NFRs has these drawbacks the advantages more than compensates the shortcomings. The adoption of NFRs places an abstraction layer for the choices and keeps the rationale about the configuration. Since there are preferences modeled, the output process will be generated taking these preferences into account. Thus, during selection of the variants the use

of NFRs also helps to reduce the variability space. Selection algorithms may rely on the information about preferences to evaluate and select the preferred variants instead of having to evaluate all variants.

The second main decision during in BVCCoN design was the use of contextual information to trigger changes in the process models. Of course the inclusion of the contextual information implied on the need of another analysis step, which increase the design time of our approach. Moreover, the context approach integrated in BVCCoN does not have a support to describe how to identify contexts. The shortcoming is the lack of a context elicitation method. However, the user of our approach may minimize this problem using different sources of information. During the preparation of the simulation study we identified that structured reports (e.g., de Wandeler (2008)) may provide valuable source for context identification. As a main advantage of the use of context in BVCCoN we highlight that it enables adaptability for business process models and helps to identify how to change the model at run-time.

6.3 Limitations

Some points are recognized limitations of our approach. In this section we discuss the most important ones.

Evolution We did not address how to deal with evolutionary issues. In the BVCCoN process described in Chapter 4 we outline a process that is mainly a top-down approach. However, the model and process associated were designed in such way that it should not be too difficult to maintain (i.e., deal with their modification). We hope that the separation among the model concepts and BVCCoN process steps will facilitate their evolution. However, this issue was not the scope of this thesis.

Complexity The several analysis steps of our approach are clearly time consuming. However, as observed during literature review (see Chapter 2) and during the comparison with concurrent work (see Section 2.5) this is a common problem for almost all approaches that deal with variability. The complexity handling is certainly an open issue that deserves further work.

The relationship between Context and NFRs In some cases the NFRs may be closely related to context. However, in our approach we do not

provide ways to represent it. An example was identified during preparation of the simulation model (see Chapter 5). The security NFR could be related with the context of security issue in the airport scenario. Unfortunately, the BVCCoN model still does not support this kind of relationship. Further investigation is welcome.

Limited manual selection The manual selection, typically used in SPL based approaches was not encouraged by BVCCoN. However, it still may be achieved if the designer considers the user intervention as a context change and models it as a context expression. So we leave to the designer of solution the burden of deciding how to deal with manual selection.

6.4 Future Work

Based on the identified limitations we can define some future work, presented as follow.

We may also investigate the use of metrics as contextual information as proposed by Oliveira et al. (2012). For example, the use of Quality of Service (QoS) metrics may be interesting for domains where process models are associated to Service oriented solutions (Oliveira et al. 2012). Some study is still necessary to see how to adapt our approach to the use of metrics. One possible solution is to try to include this information during the selection of variants, since the AHP (Saaty 2008) supports numerical values. Another solution may be to investigate if it is possible to the represent metrics as context information.

The components of our approach are focused on BPMN but other languages may also be investigated as alternatives. The BVCCoN approach could be extended to support other modeling languages such as UML Activity Diagrams (UML ADs) or Software & Systems Process Engineering Meta-Model (SPEM) (OMG 2008).

Another direction for future work consists on replicating the assessments with other domain examples. This would enable us to check its adequacy. With more assessments we can investigate in which domains the BVCCoN could be relevant and find new opportunities of improvement. As presented in Chapter 5 the application of BVCCoN in a real industrial process should take more time, which made this type of evaluation impossible given our time constraints. However, with the other methodological tools it may be possible in the future, to assess its use in situations closer to real settings. For example, De La Vara et al. (2010a) use action research to evaluate the

opinion of stakeholders about their proposal without actually putting it in practice.

In the Chapters 3 and 4 we presented some initial implementations that supported the construction of a basic model using EMF-based editor, configuration algorithms, and model transformations. These prototypes were designed as proof of concept to show that the concepts and algorithms used in our approach may work properly. However, the development of an integrated environment tool support is out of the scope of this thesis. Recognizing the current limitation of tool support we believe that the development of an integrated environment for analysis and execution of the BVCCoN approach is important. We believe that given the proper tool support our approach could be much more useful.

Further research might explore the use of pre-compiled models stored before the last step of process configuration. As presented in Section 4.6, the storage of pre-compiled process models is a possible alternative to the execution of model transformation in the last step of BVCCoN approach. It is an interesting point that may be investigated in details in the future.

More research is needed to better understand the relationship among contextual information and NFRs. As previously presented, the contextual information and NFRs may be related in some situations. For example, the activation of a context may trigger change in NFR priorities. On the other hand, the use of NFRs may create new contexts where the process may be affected, for example, increasing maintainability leads to a context of interruption in the process.

6.5 Published Material

This research led to the following publications:

- In Santos, Castro, Sánchez & Pastor (2010) we present the seminal proposal using goal models to represent variability and NFRs as mechanism of configuration for BPMN.
- In Santos, Pimentel, Castro, Sánchez & Pastor (2010) we review our approach to substitute the goal model as variability representation. In particular we introduce variants and variation points as mechanisms to represent variability, while kept the NFRs as representation of preferences.
- In order to support process configuration we identified the need to include contextual information in Santos et al. (2011). This publication introduces the notion of dynamic configuration in our approach.

- In Santos et al. (2012) we present the formalization of BVCCoN model, and the corresponding metamodel and model transformations. We also described in more detail the process configuration algorithms and a simulation-based assessment.

Appendix A

Metamodel with OCL

Listing A.1: Variability perspective metamodel

```
1 import bpmn2 : 'BPMN20.ecore#/' ;
2 import bvconcontext : 'bvcon-context.ecore#/' ;
3 import bvconnfr : 'bvcon-nfr.ecore#/' ;
4
5 package bvcon : bvcon = 'http://portal.cin.ufpe.br/ler/
    Projects/BVCCoN.aspx'
6 {
7     annotation Ecore
8     (
9         validationDelegates = 'org.eclipse.oc1.ecore.
            OCL'
10    );
11    class Variant extends bvconnfr::
        NFROperationalization
12    {
13        property varPoints#variants : VariationPoint
            [+] { ordered };
14        property contexts : bvconcontext::
            BVCCoNContext[*] { ordered };
15        property pattern#variantSource :
            WorkflowPattern[+] { ordered };
16        property part : BusinessProcessPart { ordered
            };
17        attribute isDefault : Boolean { ordered };
18        property relationships : VariantsRelationship
            [*] { ordered };
19        invariant notTheSameTarget :
20        relationships->size() > 0 implies
            relationships->forAll(rel :
                VariantsRelationship | rel.
                relationshipSource <> rel.
```

```

21         relationshipTarget);
22         invariant mustHaveName: not name.
23         oclIsUndefined();
24     }
25     class VariationPoint
26     {
27         property variants#varPoints : Variant[+] {
28             ordered };
29         attribute operator : VariationPointOperator {
30             ordered };
31         property begins : bpmn2::FlowElement {
32             ordered };
33         property ends : bpmn2::FlowElement { ordered
34             };
35         property flowElements : bpmn2::FlowElement[+]
36             { ordered };
37         attribute id : String { ordered id };
38         attribute name : String { ordered };
39     }
40     enum VariationPointOperator { serializable }
41     {
42         literal AND;
43         literal OR = 1;
44         literal XOR = 2;
45     }
46     class BusinessProcessPart
47     {
48         attribute id : String { ordered id };
49         property begins : bpmn2::FlowElement {
50             ordered };
51         property ends : bpmn2::FlowElement { ordered
52             };
53         property flowElements : bpmn2::FlowElement[+]
54             { ordered };
55     }
56     class WorkflowPattern
57     {
58         attribute wfPattern : WorkflowPatternType[?]
59             = 'WorkflowPattern.SEQUENCE' { ordered };
60         attribute id : String { ordered id };
61         property processPart : BusinessProcessPart {
62             ordered };
63         property variantSource#pattern : Variant {
64             ordered };
65     }
66     enum WorkflowPatternType { serializable }
67     {
68         literal SEQUENCE;
69         literal PARALLEL_SPLIT = 1;

```

```

57         literal SYNCHRONISATION = 2;
58         literal EXCLUSIVE_CHOICE = 3;
59         literal SIMPLE_MERGE = 4;
60         literal MULTIPLE_CHOICE = 5;
61     }
62     class Model
63     {
64         attribute id : String { ordered id };
65         property variant : Variant[+] { ordered
66             composes };
67         property variationPoints : VariationPoint[+]
68             { ordered composes };
69         property nfrs : bvconnfr::NFRModel { ordered
70             composes };
71         property contexts : bvconcontext::
72             ContextModel { ordered composes };
73         property businessProcessPart :
74             BusinessProcessPart[+] { ordered composes
75             };
76         property pattern : WorkflowPattern[+] {
77             ordered composes };
78         property variantRelationships :
79             VariantsRelationship[*] { ordered composes
80             };
81         property contributions : bvconnfr::
82             NFRSoftgoalContribution[*] { ordered
83             composes };
84     }
85     class VariantsRelationship
86     {
87         property relationshipSource : Variant {
88             ordered };
89         property relationshipTarget : Variant {
90             ordered };
91         attribute id : String { ordered id };
92     }
93     class Requires extends VariantsRelationship;
94     class Excludes extends VariantsRelationship;
95 }

```

Listing A.2: Contexts perspective metamodel

```

1 package bvconcontext : bvconcontext = 'http://portal.cin.
2   ufpe.br/ler/Projects/BVCCoN/BVCCoN-Context.aspx'
3 {
4     class ContextModel
5     {
6         attribute id : String { ordered id };

```

```

6          property contexts : BVCCoNContext[+] {
              ordered composes };
7          property contextExpressions :
              ContextExpression[+] { ordered composes };
8          property statements : Statement[*] { ordered
              composes };
9          property facts : Fact[*] { ordered composes
              };
10         property predicateDecomposition :
              PredicateDecomposition[*] { ordered
              composes };
11         property variables : Variable[*] { ordered
              composes };
12         property decompositionTypes :
              DecompositionType[*] { ordered composes };
13     }
14     class BVCCoNContext
15     {
16         property impliedBy : ContextExpression;
17         attribute name : String { ordered };
18         attribute id : String { ordered id };
19         invariant validName: not self.name.
            oclIsUndefined();
20     }
21     class ContextExpression
22     {
23         property composedBy : Predicate[+] { composes
            };
24         attribute name : String;
25         attribute id : String { ordered id };
26     }
27     abstract class Predicate;
28     class Statement extends Predicate
29     {
30         property supportedBy : Predicate;
31         attribute id : String { ordered id };
32         attribute description : String { ordered };
33         invariant NotOtherStatement:
34         supportedBy->forAll(x : Predicate | x.
            oclIsKindOf(Fact) or x.oclIsKindOf(
            PredicateDecomposition));
35     }
36     class Fact extends Predicate
37     {
38         attribute valid : Boolean;
39         attribute description : String { ordered };
40         attribute name : String { ordered id };
41         property variables : Variable[*];
42     }

```

```

43     class Variable
44     {
45         attribute name : String;
46         attribute type : String;
47         attribute value : String;
48     }
49     class PredicateDecomposition extends Predicate
50     {
51         property predicates : Predicate[*] { composes
52             };
53         property type : DecompositionType;
54         attribute name : String { ordered id };
55         invariant NotDecomposition: type.ocllsKindOf(
56             NotDecomposition) implies predicates->size
57             () = 1;
58     }
59     abstract class DecompositionType;
60     class OrDecomposition extends DecompositionType;
61     class AndDecomposition extends DecompositionType;
62     class NotDecomposition extends DecompositionType;
63 }

```

Listing A.3: NFR perspective metamodel

```

1  package bvconnfr : bvconnfr = 'http://portal.cin.ufpe.br/
2  ler/Projects/BVCCoN/BVCCoN-NFR.aspx'
3  {
4      class NFRModel
5      {
6          attribute id : String { ordered id };
7          property nfrs : NFRSoftgoal[+] { ordered
8              composes };
9          property contributions :
10             NFRSoftgoalContribution[+] { ordered
11                 composes };
12     }
13     class NFRSoftgoalPriority
14     {
15         attribute value : Integer = '1';
16     }
17     class NFRElement
18     {
19         attribute name : String;
20         attribute id : String { ordered id };
21     }
22     class NFRLink;
23     class NFRSoftgoal extends NFRElement
24     {

```

```

21         property priority : NFRSoftgoalPriority {
           composes };
22     }
23     class NFROperationalization extends NFRElement;
24     class NFRSoftgoalContribution extends NFRLink
25     {
26         property source : NFRElement;
27         property target : NFRElement;
28         attribute label : String;
29         invariant targetNFR: target.oclIsKindOf(
           NFRSoftgoal);
30         invariant andOrDecomposition:
31         if self.oclIsKindOf(AndContribution) or self.
           oclIsKindOf(OrContribution) then source.
           oclIsKindOf(NFRSoftgoal) else source.
           oclIsKindOf(NFROperationalization) endif;
32         invariant notTheSameTargetAndSource: source
           <> target;
33     }
34     class BreakContribution extends
           NFRSoftgoalContribution;
35     class MakeContribution extends
           NFRSoftgoalContribution;
36     class OrContribution extends NFRSoftgoalContribution;
37     class AndContribution extends NFRSoftgoalContribution
           ;
38     class EqualContribution extends
           NFRSoftgoalContribution;
39     class HurtContribution extends
           NFRSoftgoalContribution;
40     class HelpContribution extends
           NFRSoftgoalContribution;
41 }

```

Appendix B

Model Transformation from BVCCoN to BPMN

Listing B.1: Model Transformation with Operational QVT

```
1 modeltype BPMN uses bpmn2('http://www.omg.org/spec/BPMN
  /20100524/MODEL-XMI');
2 modeltype BVCCoN uses bvcccon('http://portal.cin.ufpe.br/ler/
  Projects/BVCCoN.aspx');
3
4 transformation WFPattern(in vp:BVCCoN, in initial:BPMN, out
  final:BPMN);
5
6 property referenceProcess:Process = initial.objectsOfType(
  Process)->asOrderedSet()->first();
7
8 main() {
9     vp.rootObjects()![Model]->map ProcessToDefinitions();
10 }
11
12
13 mapping Model::ProcessToDefinitions(): DocumentRoot@final
14 {
15
16     self.variationPoints->forEach(vpi){
17         referenceProcess := (vpi->map VPtoBPMN())->
            asOrderedSet()->first();
18     };
19
20
21     result.definitions := object Definitions{
22         rootElements += referenceProcess;
23     };
```

```

24         result.process += referenceProcess;
25
26     }
27
28     mapping VariationPoint::VPtoBPMN():Process
29     disjuncts VariationPoint::applyParallelPattern,
30     VariationPoint::applySequencePattern,
31     VariationPoint::applyMultiChoicePattern,
32     VariationPoint::applyExclusiveChoicePattern,
33     VariationPoint::applySequenceDefaultPattern {
34     }
35
36     mapping VariationPoint::applySequenceDefaultPattern():
37         Process
38     when {
39         (self.variants.pattern.wfPattern->includes(
40             WorkflowPattern::PARALLE_SPLIT) or
41             self.variants.pattern.wfPattern->includes(
42                 WorkflowPattern::EXCLUSIVE_CHOICE) or
43             self.variants.pattern.wfPattern->includes(
44                 WorkflowPattern::MULTIPLE_CHOICE)) and
45             (self.variants->size() = 1) and
46             (self.begins <> self.ends)
47     }
48     {
49         init {
50             result := referenceProcess;
51         }
52         var begins := self.begins;
53         var ends := self.ends;
54
55         var variant := self.variants->first();
56
57         if (begins.oclIsUndefined() = false)
58         then {
59             var seq = new SequenceFlow();
60             seq.sourceRef := begins.oclAsType(FlowNode);
61             seq.targetRef := variant.oclAsType(FlowNode)
62             ;
63             result.flowElements += seq;
64             var seq2 = new SequenceFlow();
65             seq2.sourceRef := variant.oclAsType(FlowNode
66                 );
67             seq2.targetRef := ends.oclAsType(FlowNode);
68             result.flowElements += seq2;
69         }
70     }
71     endif;

```

```

67
68 }
69
70 mapping VariationPoint::applyParallelPattern(): Process
71 when{
72     (self.operator = VariationPointOperator::AND) and
73     (self.variants.pattern.wfPattern->includes(
74         WorkflowPattern::PARALLE_SPLIT)) and
75     (self.variants.pattern.wfPattern->count(
76         WorkflowPattern::PARALLE_SPLIT)>=2)
77     and (self.begins <> self.ends)
78 }
79 {
80     init {
81         result := referenceProcess;
82     }
83     var begins := self.begins;
84     var ends := self.ends;
85
86     var variants := self.variants;
87     if (begins.ocIsUndefined() = false)
88     then {
89         var split = new ParallelGateway();
90         var join = new ParallelGateway();
91
92         result.flowElements += split;
93         result.flowElements += join;
94
95         var beginSplitSeq = new SequenceFlow();
96         beginSplitSeq.sourceRef := begins.ocAsType(
97             FlowNode);
98         beginSplitSeq.targetRef := split.ocAsType(
99             FlowNode);
100
101         result.flowElements += beginSplitSeq;
102
103         var joinEndSeq = new SequenceFlow();
104         joinEndSeq.sourceRef := join.ocAsType(
105             FlowNode);
106         joinEndSeq.targetRef := ends.ocAsType(
107             FlowNode);
108
109         result.flowElements += joinEndSeq;
110         variants->forEach(v){
111             var splitVarBeginSeq = new
112                 SequenceFlow();
113             splitVarBeginSeq.sourceRef := split.
114                 ocAsType(FlowNode);

```

```

108             splitVarBeginSeq.targetRef := v.part
109                 .begins.oclAsType(FlowNode);
110             result.flowElements +=
111                 splitVarBeginSeq;
112             var varEndJoinSeq = new SequenceFlow
113                 ();
114             varEndJoinSeq.sourceRef := v.part.
115                 ends.oclAsType(FlowNode);
116             varEndJoinSeq.targetRef := join.
117                 oclAsType(FlowNode);
118             result.flowElements += varEndJoinSeq;
119         }
120     }
121     endif;
122 }
123
124 mapping VariationPoint::applySequencePattern(): Process
125 when {
126     (self.operator = VariationPointOperator::AND) and
127     (self.variants.pattern.wfPattern->includes(
128         WorkflowPattern::SEQUENCE)) and
129     (self.begins <> self.ends)
130 }
131 {
132     init {
133         result := referenceProcess;
134     }
135     var begins := self.begins;
136     var ends := self.ends;
137     var variants := self.variants;
138
139     var head := begins;
140     var tail := ends;
141
142     variants->forEach (v){
143         if (begins.oclIsUndefined() = false)
144             then {
145                 tail := v.part.begins;
146
147                 var seq = new SequenceFlow();
148                 seq.sourceRef := head.oclAsType(
149                     FlowNode);
150                 seq.targetRef := tail.oclAsType(
151                     FlowNode);
152
153                 head := v.part.ends;
154                 result.flowElements += seq;
155             }
156     }
157 }

```

```

149             endif;
150         };
151
152
153         var seq = new SequenceFlow();
154         seq.sourceRef := head.oclAsType(FlowNode);
155         seq.targetRef := ends.oclAsType(FlowNode);
156         result.flowElements += seq;
157
158
159     }
160
161     mapping VariationPoint::applyMultiChoicePattern(): Process
162     when {
163         (self.operator = VariationPointOperator::OR) and
164         (self.variants.pattern.wfPattern->includes(
165             WorkflowPattern::MULTIPLE_CHOICE)) and
166         (self.begins <> self.ends)
167     {
168         init {
169             result := referenceProcess;
170         }
171         var begins := self.begins;
172         var ends := self.ends;
173         var variants := self.variants;
174
175
176         var t : Set(FlowElement);
177         t += begins;
178         t += ends;
179
180         var be: Set(FlowElement) := referenceProcess.
181             flowElements - (t->intersection(referenceProcess.
182                 flowElements));
183
184         if (begins.oclIsUndefined() = false)
185         then {
186             var split = new InclusiveGateway();
187             var join = new InclusiveGateway();
188             result.flowElements += split;
189             result.flowElements += join;
190
191             var beginSplitSeq = new SequenceFlow();
192             beginSplitSeq.sourceRef := begins.oclAsType(
193                 FlowNode);
194             beginSplitSeq.targetRef := split.oclAsType(
195                 FlowNode);

```

```

193         result.flowElements += beginSplitSeq;
194
195         var joinEndSeq = new SequenceFlow();
196         joinEndSeq.sourceRef := join.oclAsType(
            FlowNode);
197         joinEndSeq.targetRef := ends.oclAsType(
            FlowNode);
198         result.flowElements += joinEndSeq;
199
200         variants->forEach(v){
201             var splitVarBeginSeq = new
                SequenceFlow();
202             splitVarBeginSeq.sourceRef := split.
                oclAsType(FlowNode);
203             splitVarBeginSeq.targetRef := v.part
                .begins.oclAsType(FlowNode);
204             result.flowElements +=
                splitVarBeginSeq;
205             var varEndJoinSeq = new SequenceFlow
                ();
206             varEndJoinSeq.sourceRef := v.part.
                ends.oclAsType(FlowNode);
207             varEndJoinSeq.targetRef := join.
                oclAsType(FlowNode);
208             result.flowElements += varEndJoinSeq;
209         }
210     }
211     endif;
212 }
213
214 mapping VariationPoint::applyExclusiveChoicePattern():
    Process
215 when {
216     (self.operator = VariationPointOperator::XOR) and
217     (self.variants.pattern.wfPattern->includes(
        WorkflowPattern::EXCLUSIVE_CHOICE)) and
218     (self.begins <> self.ends)
219 }
220 {
221     init {
222         result := referenceProcess;
223     }
224     var begins := self.begins;
225     var ends := self.ends;
226     var variants := self.variants;
227
228     if (begins.oclIsTypeOf(FlowElement))
229     then {
230         var split = new ExclusiveGateway();

```

```

231         var join = new ExclusiveGateway();
232
233         result.flowElements += split;
234         result.flowElements += join;
235
236         var beginSplitSeq = new SequenceFlow();
237         beginSplitSeq.sourceRef := begins.oclAsType(
238             FlowNode);
239         beginSplitSeq.targetRef := split.oclAsType(
240             FlowNode);
241         result.flowElements += beginSplitSeq;
242
243         var joinEndSeq = new SequenceFlow();
244         joinEndSeq.sourceRef := join.oclAsType(
245             FlowNode);
246         joinEndSeq.targetRef := ends.oclAsType(
247             FlowNode);
248         result.flowElements += joinEndSeq;
249
250         variants->forEach(v){
251             var splitVarBeginSeq = new
252                 SequenceFlow();
253             splitVarBeginSeq.sourceRef := split.
254                 oclAsType(FlowNode);
255             splitVarBeginSeq.targetRef := v.part
256                 .begins.oclAsType(FlowNode);
257             result.flowElements +=
258                 splitVarBeginSeq;
259             var varEndJoinSeq = new SequenceFlow
260                 ();
261             varEndJoinSeq.sourceRef := v.part.
262                 ends.oclAsType(FlowNode);
263             varEndJoinSeq.targetRef := join.
264                 oclAsType(FlowNode);
265             result.flowElements += varEndJoinSeq;
266         }
267     }
268 }
endif;
}

```

Appendix C

Airport Simulation Scenario - BPMN Models

This Appendix contains the BPMN models used in the simulation described in Chapter 5.

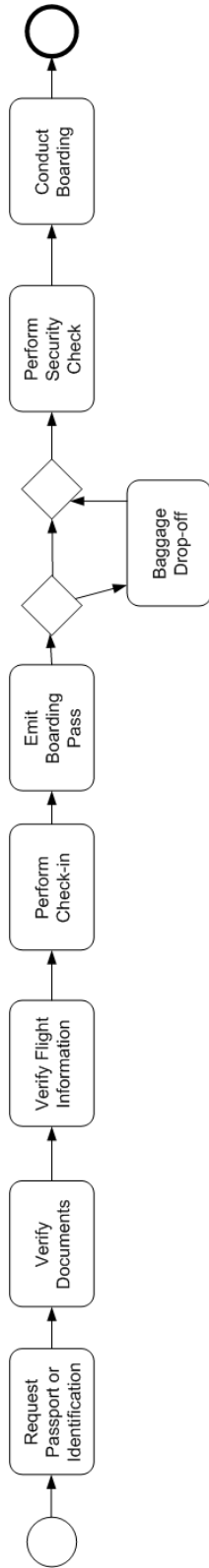


Figure C.1: Reference model

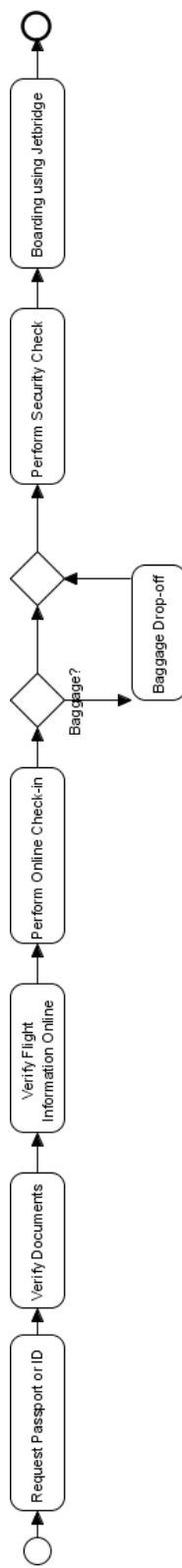


Figure C.2: BPMN Process with priority for Performance NFR

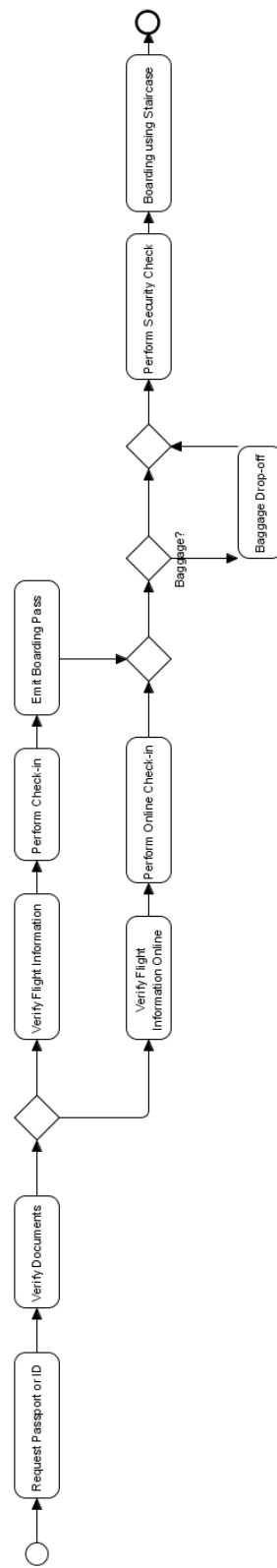


Figure C.3: BPMN Process with priority for Reliability NFR

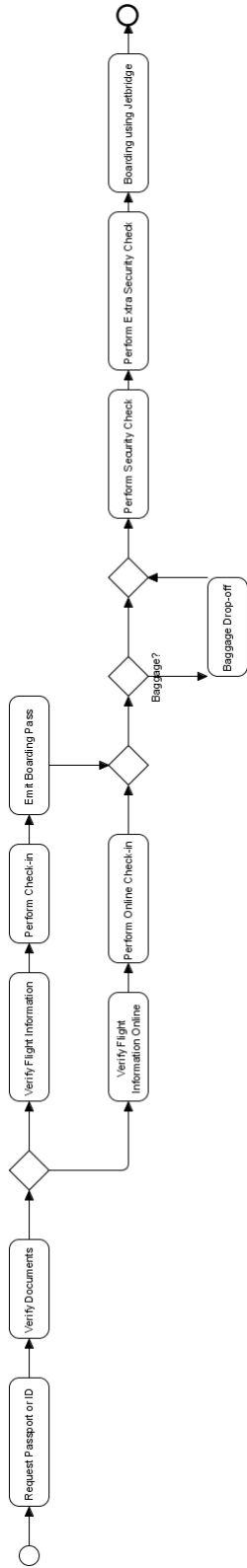


Figure C.4: BPMN Process with priority for Security NFR

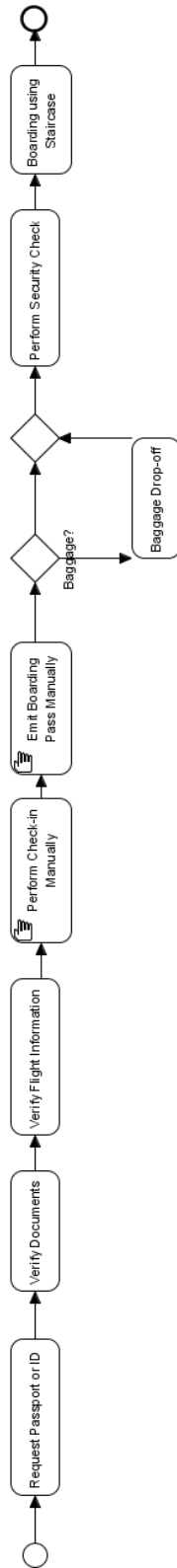


Figure C.5: BPMN Process for Without Internet Context

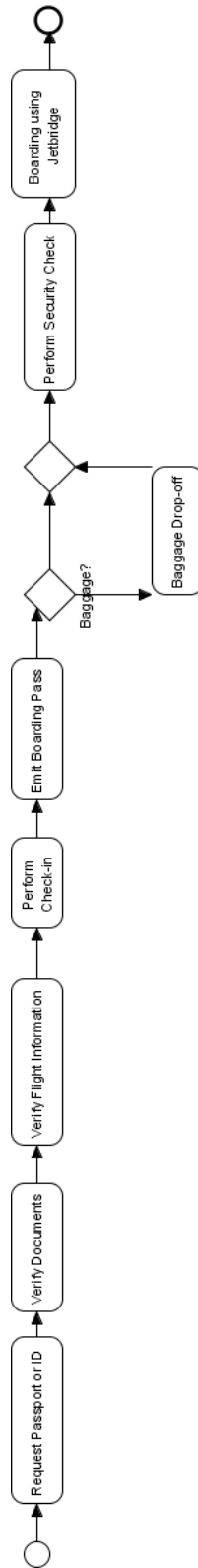


Figure C.6: BPMN Process for Without Online Check-in Context

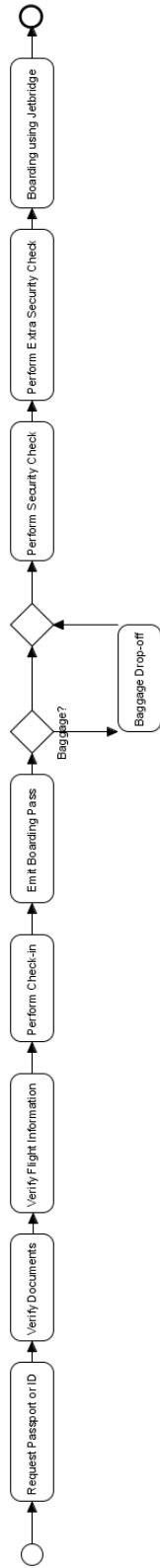


Figure C.7: BPMN Process for Security Issue Context

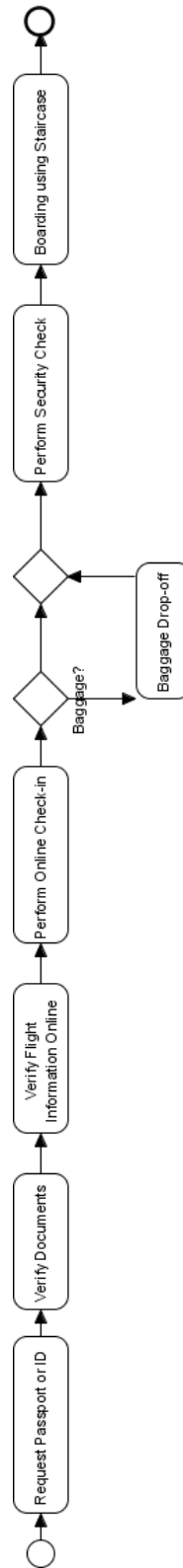


Figure C.8: BPMN Process for Small Airport Context

Appendix D

Airport Simulation Scenario - Contexts and NFRs Models

This Appendix contains the information used to design the simulation scenarios from Chapter 5.

Table D.1: Relationship between Tasks and Resources

Tasks	Resources							
	Jet bridge	Internet service	Security Staff	X-Ray	Check-in staff	Metal detector	Boarding Staff	Staircase
Perform Check-in		1			1			
Perform Check-in Manually					1			
Perform Online Check-in		1						
Boarding Using Staircase							5	1
Delay boarding							1	
Boarding using Jetbridge	1						3	
Perform Security Check			1	1		1		
Perform Extra Security Check			2	1		1		
Verify Flight information		1						
Request Visa/Documents					1			
Verify Documents					1			
Baggage Drop-off					1			
Emit boarding pass					1			
Verify Flight information On line		1						

Table D.2: Relationship between Tasks and Active Contexts

	Active Context			
	Without online check-in	Without Internet service	Security Alert	Small Airport
Valid Tasks				
Perform Check-in		invalid		
Perform Check-in Manually				
Perform Online Check-in	invalid	invalid		
Boarding Using Staircase				
Delay boarding	invalid			
Boarding using Jetbridge				invalid
Perform Security Check				
Perform Extra Security Check	invalid	invalid		invalid
Verify Flight information				
Request Visa/Documents				
Verify Documents				
Baggage Drop-off				
Emit boarding pass		invalid		
Verify Flight information On line	invalid	invalid		

Table D.3: Priorities to NFRs

	NFRs		Priority NFR		
			Performance	Reliability	Security
Performance	Time Performance	Throughput	9	8	4
		Response Time	8	7	3
Reliability	Availability		7	10	6
	Accuracy		6	9	5
Security	Confidentiality	Internal Confidentiality	5	5	8
		External Confidentiality	4	6	9
	Access Control		3	4	10

References

- Aburub, F., Odeh, M. & Beeson, I. (2007), ‘Modelling non-functional requirements of business processes’, *Information and Software Technology* **49**(11-12), 1162–1171.
- Aguilar-Savén, R. S. (2004), ‘Business process modelling : Review and framework’, *International Journal of Production Economics* **90**, 129–149.
- Akehurst, D. & Kent, S. (2002), A relational approach to defining transformations in a metamodel, in ‘UML 2002 — The Unified Modeling Language’, Vol. 2460 of *Lecture Notes in Computer Science*, Springer, Dresden, Germany.
- Aleixo, F., Freire, M., Alencar, D., Campos, E. & Kulesza, U. (2012), A Comparative Study of Compositional and Annotative Modelling Approaches for Software Process Lines, in ‘Proceedings of the 26th Brazilian Symposium on Software Engineering (SBES 2012)’, number ii, Ieee, Natal, Brasil, pp. 51–60.
URL: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6337893>
- Ali, R., Dalpiaz, F. & Giorgini, P. (2010), ‘A goal-based framework for contextual requirements modeling and analysis’, *Requirements Engineering* **15**, 439–458.
- Bachmann, F. & Clements, P. C. (2005), Variability in software product lines. Technical Report CMU/SEI-2005-TR012, Technical Report September, Software Engineering Institute, Pittsburgh, USA.
- Bai, Y., Yang, J. & Qiu, Y. (2008), OntoCBR: Ontology-Based CBR in Context-Aware Applications, in ‘Proceedings of the International Conference on Multimedia and Ubiquitous Engineering (MUE 2008)’, IEEE Comput. Soc. Press, pp. 164–169.

- Batory, D. S. (2005), Feature Models, Grammars and Propositional Formulas, *in* J. H. Obbink & K. Pohl, eds, ‘Proceedings of the 9th International Conference on Software Product Lines (SPLC’05)’, Springer, pp. 7–20.
- Batory, D. S. & Geraci, B. J. (1997), ‘Composition Validation and Subjectivity in GenVoca Generators.’, *IEEE Transactions on Software Engineering* **23**(2), 67–84.
- Bayer, J., Buhl, W., Giese, C., Lehner, T., Ocampo, A., Puhlmann, F., Richter, E., Schnieders, A., Weiland, J. & Weske, M. (2005), Process Family Engineering: Modeling variant-rich processes, Technical report, PESOA -Process Family Engineering in Service-Oriented Applications, BMBF-Project.
- Becker, J., Kugeler, M. & Rosemann, M. (2003), *Process management: a guide for the design of business processes*, Springer.
- Bentellis, A. & Boufaïda, Z. (2008), ‘Conceptual Method for Flexible Business Process’, *World Academy of Science, Engineering and Technology* **37**, 286–290.
- Berczuk, B. S. P. & Appleton, B. (2002), *Software Configuration Management Patterns : Effective Teamwork , Practical Integration*.
- Bessai, K., Claudepierre, B., Saidani, O. & Nurcan, S. (2008), Context-aware business process evaluation and redesign, *in* ‘Proceedings of the 9th Workshop on Business Process Modeling, Development and Support - BPMDS 2008’, Springer, Montpellier, France, pp. 1–10.
- Bettini, C., Brdiczka, O., Henriksen, K., Indulska, J., Nicklas, D., Ranganathan, A. & Riboni, D. (2010), ‘A Survey of Context Modelling and Reasoning Techniques’, *Pervasive and Mobile* **6**, 161–180.
- Boehm, B., Brown, J. R., Kaspar, H., Lipow, M., MacLeod, G. & Merritt, M. J. (1978), *Characteristics of Software Quality*, North-Holland Publishing Co.
- Cardoso, E. C. S., Almeida, J. P. A., Guizzardi, G. & Guizzardi, R. S. S. (2009), Eliciting Goals for Business Process Models with Non-Functional Requirements Catalogues, *in* ‘Enterprise, Business-Process and Information Systems Modeling: Proceedings of the 10th International Workshop, Bpmds 2009, and 14th International Conference, Emmsad 2009, Held at Caise 2009 June 8-9, 2009’, Springer, Amsterdam, the Netherlands, p. 33.

- Cetina, C., Giner, P., Fons, J. & Pelechano, V. (2009), ‘Autonomic Computing through Reuse of Variability Models at Runtime: The Case of Smart Homes’, *Computer* **42**, 37–43.
- Chang, J. F. (2006), *Business process management systems: strategy and implementation*, Auerbach Publications, Baton Rouge, USA.
- Chatel, P., Leger, A. & Lockerbie, J. (2011), Deliverable D6.1 Requirements and scenarios for the ”Passenger-friendly airport”, Technical report, CHOReOS Project.
URL: www.choreos.eu
- Chung, L., Nixon, B. A., Yu, E. & Mylopoulos, J. (2000), *Non-Functional Requirements in Software Engineering*, Vol. 5, Kluwer Academic Publishers.
- Cortez, P. & Embrechts, M. (2011), Opening black box data mining models using sensitivity analysis, *in* ‘Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM), 2011’, Ieee, pp. 341–348.
- Cysneiros, L. M. & Leite, J. C. S. D. P. (1998), Requisitos Não funcionais como validadores de modelos de dados, *in* ‘Anais do I Workshop de Engenharia de Requisitos’, Maringá, Brasil.
- Czarnecki, K. & Eisenecker, U. W. (2000), *Generative Programming: methods tools and applications*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA.
- Czarnecki, K., Helsen, S. & Eisenecker, U. W. (2005), ‘Formalizing Cardinality-Based Feature Models and Their Specialization’, *Software Process: Improvement and Practice* **10**(1), 7–29.
- Dalpiaz, F., Ali, R., Asnar, Y., Bryl, V. & Giorgini, P. (2008), Applying Tropos to Socio-Technical System Design and Runtime Configuration, *in* ‘Proceeding of the Italian workshop Dagli OGGETTI agli AGENTI (WOA’08)’, Palermo, Italy.
- Davis, A. M. (1993), *Software Requirements: Objects, Functions and States*, Prentice-Hall.
- De La Vara, J. L., Ali, R., Dalpiaz, F., Sánchez, J. & Giorgini, P. (2010a), Business Processes Contextualisation via Context Analysis, *in* ‘Proceedings of the ER 2010’, pp. 1–6.

- De La Vara, J. L., Ali, R., Dalpiaz, F., Sanchez, J. & Giorgini, P. (2010b), COMPRO : A Methodological Approach for Business Process Contextualisation, *in* 'Proceedings of the 18th International Conference on Cooperative Information Systems'.
- de Wandeler, Y. (2008), Planning for Delay : influence of flight scheduling on airline punctuality, Technical report, EUROCONTROL.
URL: www.eurocontrol.int/prc/gallery/content/public/PRR_2008.pdf
- Dumas, M., van Der Aalst, W. M. P. & Hofstede, A. H. M. T. (2005), *Process-Aware Information Systems: Bridging People and Software through Process Technology.*, John Wiley & Sons.
- Eclipse.org (2011), 'BPMN 2.0 Metamodel'.
URL: <http://www.eclipse.org/modeling/mdt/?project=bpmn2>
- Eclipse.org (2012), 'Eclipse Modeling Framework Project (EMF)'.
URL: <http://www.eclipse.org/modeling/emf/>
- Figueira, J., Greco, S. & Ehrgott, M. (2005), *Multiple criteria decision analysis: state of the art surveys*, Vol. 78, Kluwer Academic Publishers, London, UK.
- Fowler, K. (2004), 'Mission-critical and safety-critical development', *IEEE Instrumentation and Measurement Magazine* **7**(4), 52–59.
URL: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1383466>
- González-Baixauli, B., Laguna, M. A. & Leite, J. C. S. D. P. (2007), Using Goal-Models to Analyze Variability, *in* 'Proceedings of the 1st International Workshop on Variability Modelling of Software-intensive Systems, VAMOS 2007'.
- Gottschalk, F., van Der Aalst, W. M. P., Jansen-Vullers, M. H. & La Rosa, M. (2008), 'Configurable Workflow Models', *International Journal of Cooperative Information Systems* **17**(02), 177.
- Greenwood, D. & Rimassa, G. (2007), Autonomic Goal-Oriented Business Process Management, *in* 'Proceedings of the 3rd International Conference on Autonomic and Autonomous Systems (ICAS'07)', Ieee, pp. 43–49.
- Gregoriades, A., Sutcliffe, A. & Shin, J. (2003), 'Assessing the reliability of socio-technical systems', *Systems engineering* **6**(3), 210–223.

- Hallerbach, A., Bauer, T. & Reichert, M. (2008), Context-based Configuration of Process Variants, *in* ‘Proceedings of the 3rd International Workshop on Context-Aware Business Process Management (TCoB’08)’, number Activity 1, Barcelona, Spain, pp. 31–40.
- Hallerbach, A., Bauer, T. & Reichert, M. (2009), ‘Capturing variability in business process models: the Provop approach’, *Software Process: Improvement and Practice* **22**(6-7), 519–546.
- Henricksen, K. & Indulska, J. (2004), A software engineering framework for context-aware pervasive computing, *in* ‘Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications, 2004.’, Ieee, pp. 77–86.
- Horkoff, J. & Yu, E. (2010), Finding Solutions in Goal Models: An Interactive Backward Reasoning Approach, *in* ‘Conceptual Modeling-Er 2010: 29th International Conference on Conceptual Modeling, Vancouver, Bc, Canada, November 1-4, 2010, Proceedings’, Springer-Verlag New York Inc, p. 59.
- IEEEStd-830 (1998), IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications, Technical Report October, IEEE Computer Society.
- Jordan, D. & Evdemon, J. (2007), Web Services Business Process Execution Language Version 2.0, Technical report.
- Kang, K. C., Cohen, S., Hess, J., Novak, W. & Peterson, A. (1990), Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI- 90-TR-21, Technical report, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Unites States.
- Kirner, T. G. & Davis, A. M. (1996), ‘Requirements specification of real-time systems: temporal parameters and timing-constraints’, *Information and Software Technology* **38**(12), 735–741.
- Kitchenham, B., Charters, S., Budgen, D., Brereton, P., Turner, M., Linkman, S., Jorgensen, M., Mendes, E. & Visaggio, G. (2007), Guidelines for performing Systematic Literature Reviews in Software Engineering, Technical report, School of Computer Science and Mathematics, Keele University, UK.
- Kotonya, G. & Sommerville, I. (1998), *Requirements Engineering: Processes and Techniques*, Wiley, John & Sons Inc.

- Krikhaar, R., Mosterman, W., Veerman, N. & Verhoef, C. (2009), ‘Enabling system evolution through configuration management on the hardware/-software boundary’, *Systems Engineering* **12**(3), 233–264.
- La Rosa, M. (2009), Managing Variability in Process-Aware Information Systems, Phd thesis, Queensland University of Technology.
- La Rosa, M., van Der Aalst, W. M. P., Dumas, M. & ter Hofstede, A. H. M. (2009), ‘Questionnaire-based variability modeling for system configuration’, *Software and Systems Modeling* **8**(2), 251–274.
- Lapouchnian, A., Yu, Y. & Mylopoulos, J. (2007), Requirements-Driven Design and Configuration Management of Business Processes, *in* ‘Proceedings of the 5th International Conference on Business Process Management, BPM 2007’, pp. 246–261.
- Le, M. S. T. (2009), A Simulation Framework for Socio-Technical Systems, Master thesis, European Master in Informatics.
- Leggat, T., Yven, C. & Wandeler, Y. D. (2009), A Matter of Time : Air Traffic Delay in Europe, Technical report, EUROCONTROL.
- Liaskos, S., Jalman, R. & Aranda, J. (2012), On eliciting contribution measures in goal models, *in* ‘Proceedings of the 20th IEEE International Requirements Engineering Conference (RE 2012)’, IEEE, pp. 221–230.
- Liaskos, S., Jiang, L., Lapouchnian, A., Wang, Y., Yu, Y., Leite, J. C. S. D. P. & Mylopoulos, J. (2007), Exploring the Dimensions of Variability: a Requirements Engineering Perspective, *in* ‘Proceedings of the 1st International Workshop on Variability Modelling of Software-intensive Systems, VAMOS 2007’.
- Liaskos, S., Lapouchnian, A., Yu, Y., Yu, E. & Mylopoulos, J. (2006), On Goal-based Variability Acquisition and Analysis, *in* ‘Proceedings of the 14th IEEE International Requirements Engineering Conference, RE 2006’, pp. 92–96.
- Maiden, N. A. M., Jones, S. V., Manning, S., Greenwood, J. & Renou, L. (2004), Model-Driven Requirements Engineering : Synchronising Models in an Air Traffic Management Case Study, *in* ‘Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE 2004)’, Springer - Verlag Berlin Heidelberg.

- Mairiza, D., Zowghi, D. & Nurmuliani, N. (2010), An Investigation into the Notion of Non-Functional Requirements, *in* ‘Proceedings of the 25th Annual ACM Symposium on Applied Computing - SAC-ACM 2010’, pp. 311–317.
- Mannion, M. (2002), Using First-Order Logic for Product Line Model Validation., *in* ‘Proceedings of the 2nd International Conference on Software Product Lines (SPLC’02), volume 2379 of Lecture Notes in Computer Science’, Springer, pp. 176–187.
- Maté, J. L. & Silva, A. (2005), *Requirements Engineering for Sociotechnical Systems*, Information Science Publishing, Lodon.
- McKinley, P. K., Sadjadi, S. M., Kasten, E. P. & Cheng, B. H. C. (2004), ‘Composing adaptive software’, *Computer* **37**(7), 56–64.
- Montero, I. (2009), A Methodology Fragment for Developing Families of Business Information Systems: Improving the Design of Business Families for Service Oriented Architectures, Technical report, Universidad de Servilla.
- Montero, I., Peña, J. & Ruiz-Cortés, A. (2007), Business Family Engineering: Does it make sense?, *in* ‘Proceedings of the 1st Taller sobre Procesos de Negocio e Ingeniería del Software (PNIS 2007)’.
- Montero, I., Peña, J. & Ruiz-Cortés, A. (2008a), From feature models to business processes, *in* ‘Proceedings of the IEEE International Conference on Services Computing’, Vol. 2, pp. 605–608.
- Montero, I., Peña, J. & Ruiz-Cortés, A. (2008b), Representing Runtime Variability in Business-Driven Development Systems, *in* ‘Proceedings of the Seventh International Conference on Composition-Based Software Systems (ICCBSS08)’, IEEE Computer Society Press, pp. 605–608.
- Mylopoulos, J., Chung, L. & Nixon, B. A. (1992), ‘Representing and using non-functional requirements: A process-Oriented approach’, *IEEE Transactions on Software Engineering* **18**(6), 483–497.
- Northrop, L. M. (2002), ‘Sei’s software product line tenets’, *IEEE Software* **19**(4), 32–40.
- Nunes, V. T., Santoro, F. M. & Borges, M. R. (2009), ‘A context-based model for Knowledge Management embodied in work processes’, *Information Sciences* **179**(15), 2538–2554.

- Oliveira, K., Castro, J., Santos, E., Fidalgo, R., España, S. & Pastor, O. (2012), A Multi Level approach to Autonomic Business Process, *in* 'Proceedings of the 26th Brazilian Symposium on Software Engineering (SBES 2012)', IEEE Computer Society, Natal, Brasil.
- OMG, O. M. G. (2006), Diagram Interchange, Technical Report April, Object Management Group.
URL: <http://www.omg.org/spec/UMLDI/1.0/PDF/>
- OMG, O. M. G. (2008), Software & Systems Process Engineering Meta-Model Specification, Technical Report April, Object Management Group.
URL: <http://www.omg.org/spec/SPEM/2.0/PDF/>
- OMG, O. M. G. (2009), OMG Unified Modeling Language TM (OMG UML), Infrastructure, Technical Report February, Object Management Group, Inc.
- OMG, O. M. G. (2010a), Business Process Model and Notation (BPMN) - Specification v2.0, Technical Report January, Object Management Group, Inc. (OMG).
URL: <http://www.omg.org/spec/BPMN/2.0/>
- OMG, O. M. G. (2010b), Object Constraint Language, Technical Report February.
- Pavlovski, C. J. & Zou, J. (2008), Non-functional requirements in business process modeling, *in* 'Proceedings of the fifth on Asia-Pacific conference on conceptual modelling (APCCM '08)', Australian Computer Society, Inc., pp. 103–112.
- Pereira, T. C., Alencar, F., Silva, J. & Castro, J. (2013), Requisitos Não-Funcionais em Modelos de Processos de Negócio : Uma Revisão Sistemática, *in* 'Proceedings of 9th Brazilian Symposium on Information Systems (SBSI 2013)', SBC Press, João Pessoa, Brasil.
- Ploesser, K., Janiesch, C., Recker, J. & Rosemann, M. (2009), Context Change Archetypes : Understanding the Impact of Context Change on Business Processes, *in* 'Proceedings of the 20th Australasian Conference on Information Systems - ACIS 2009'.
- Ploesser, K., Peleg, M., Soffer, P., Rosemann, M. & Recker, J. (2009), 'Learning from Context to Improve Business Processes', *BPTrends* 6, 1–7.

- Pohl, K., Bockle, G. & Linden, F. J. V. D. (2005), *Software product line engineering*, Vol. 49, Springer - Verlag Berlin Heidelberg.
- Puhlmann, F., Schnieders, A., Weiland, J. & Weske, M. (2005), Variability Mechanisms for Process Models, Technical report, PESOA - Process Family Engineering in Service-Oriented Applications, BMBF-Project.
- Raines, S. (2011), Crisis Management Plan, Technical Report March, University of Memphis, Memphis.
- Reichert, M. & Weber, B. (2012), *Enabling Flexibility in Process-Aware Information Systems*, number c, Springer Berlin Heidelberg, Berlin, Heidelberg.
URL: <http://www.springerlink.com/index/10.1007/978-3-642-30409-5>
- Rosemann, M., Recker, J. & Flender, C. (2008), 'Contextualisation of business processes', *International Journal of Business Process Integration and Management* **3**(1), 47.
- Russell, N., van Der Aalst, W. M. P., ter Hofstede, A. H. M. & Wohed, P. (2006), On the suitability of UML 2.0 activity diagrams for business process modelling, in 'Proceedings of the Third Asia-Pacific Conference on Conceptual modelling', Vol. 53, Australian Computer Society, Hobart, Australia.
- Saaty, T. (2005), 'Making and validating complex decisions with the AHP/ANP', *Journal of Systems Science and Systems Engineering* **14**(1), 1–36.
- Saaty, T. L. (2008), 'Relative Measurement and Its Generalization in Decision Making Why Pairwise Comparisons are Central in Mathematics for the Measurement of Intangible Factors The Analytic Hierarchy/Network Process', *Rev. R. Acad. Cien.* **102**(2), 251–318.
- Santos, E., Castro, J., Sánchez, J. & Pastor, O. (2010), A Goal-Oriented Approach for Variability in BPMN, in G. Hadad, O. Dieste & J. P. Carvalho, eds, 'Proceedings of the 13th Workshop on Requirement Engineering, WER 2010', Cuenca, Ecuador, pp. 17–28.
- Santos, E., Pimentel, J., Castro, J. & Finkelstein, A. (2012), On the Dynamic Configuration of Business Process Models, in 'Enterprise, Business-Process and Information Systems Modeling. , Proceedings of the 13th International Workshop, BPMDS 2012, and 17th International Conference,

- EMMSAD/EuroSymposium 2012, held at CAiSE 2012, Gdansk, Poland', Gdansk, Poland, pp. 331–346.
- Santos, E., Pimentel, J., Castro, J., Sánchez, J. & Pastor, O. (2010), Configuring the Variability of Business Process Models Using Non-Functional Requirements, *in* I. Bider, T. Halpin, J. Krogstie, S. Nurcan, E. Proper, R. Schmidt & R. Ukor, eds, 'Enterprise, Business-Process and Information Systems Modeling. , Proceedings of the 11th International Workshop, BPMDS 2010, and 15th International Conference, EMMSAD 2010, held at CAiSE 2010, Hammamet, Tunisia, June 7-8, 2010', Springer Berlin Heidelberg, Hammamet, Tunisia, pp. 274–286.
- Santos, E., Pimentel, J., Dermeval, D., Castro, J. & Pastor, O. (2011), Using NFR and Context to Deal with Adaptability in Business Process Models, *in* 'Proceedings of the 2nd International Workshop on Requirements@Runtime 2011', Trento, Italy.
- Schnieders, A. & Puhlmann, F. (2006), Variability Mechanisms in E-Business Process Families, *in* 'Proceedings of the 9th International Conference on Business Information Systems, BIS 2006'.
- Soffer, P. & Wand, Y. (2005), 'On the notion of soft-goals in business process modeling', *Business Process Management Journal* **11**(6), 663–679.
- Sommerville, I. & Sawyer, P. (1997), *Requirements engineering: a good practice guide*, John Wiley & Sons.
- Suleiman, B., Tasic, V., Jeffery, R. & Liu, J. (2009), 'Models and algorithms for business value-driven adaptation of business processes and software infrastructure', *Proceedings of the 31st International Conference on Software Engineering* pp. 387–390.
- van Der Aalst, W. M. P. (2011), *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, 1st editio edn, Springer.
- van Der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B. & Barros, A. (2003), 'Workflow patterns', *Distributed and parallel databases* **14**(1), 5–51.
- Wernick, P. & Hall, T. (2007), Getting the best out of software process simulation and empirical research in software engineering, *in* 'Proceedings of the 2nd International Workshop on Realising Evidence-Based Software Engineering (REBSE 2007)'.

- White, S. A. & Miers, D. (2008), *BPMN Modeling and Reference Guide: Understanding and Using BPMN*, Future Strategies Book Division.
- Wohed, P., van Der Aalst, W. M. P., Dumas, M., ter Hofstede, A. H. M. & Russell, N. (2005), Pattern-based Analysis of BPMN, Technical report, Queensland University of Technology, Brisbane, Australia.
- Wohlin, C., Runeson, P., Host, M., Ohlsson, M., Regnell, B. & Wesslen, A. (2000), *Experimentation in Software Engineering*, Kluwer Academic Publishers, London, UK.
- Wooldridge, M. (2002), *An Introduction to MultiAgent Systems*, John Wiley & Sons, Chichester, England.
- Xavier, L., Alencar, F., Castro, J. & Pimentel, J. (2010), Integração de Requisitos Não-Funcionais a Processos de Negócio: Integrando BPMN e NFR, *in* G. Hadad, O. Dieste & J. P. Carvalho, eds, 'Proceedings of the 13th Workshop on Requirement Engineering, WER 2010', Cuenca, Equador, pp. 29–40.
- Yu, E. (1995), Modelling Strategic Relationships for Process Reengineering, Phd thesis, University of Toronto.
- Yu, Y., Lapouchnian, A., Liaskos, S., Mylopoulos, J. & Leite, J. C. S. D. P. (2008), From goals to high-variability software design, *in* 'Foundations of Intelligent Systems, Proceedings of the 17th International Symposium in Methodologies for Intelligent Systems, ISMIS 2008', Vol. 4994, Springer, Toronto, Canada, pp. 1–16.