



Faculty of Engineering and Technology

Master Program of Computing

**A Systematic Business Model-Driven Approach for
Deriving UML-Based Requirement Specifications**

نهج منظم يعتمد على نماذج الاعمال لإشتقاق متطلبات النظام ممثلة بلغة
التصميم الموحدة

Author:

Salam Turkman

Supervisor:

Dr. Adel Taweel

Committee:

Dr. Adel Taweel

Dr. Derar Elayyan

Dr. Mohanad Jabari

This Thesis was submitted in partial fulfillment of the requirements for the
Master's Degree in Computing from the Faculty of Graduate Studies at
Birzeit University, Palestine

28/1/2017



**A Systematic Business Model-Driven Approach for Deriving UML-
Based Requirement Specifications**

By Salam Turkman

Approved by the thesis committee

Dr. Adel Taweel, Birzeit University

Dr. Derar Eleyan, Palestine Technical University- Kadoorie

Dr. Mohanad Jabari, Hebron University

Date Approved

A Systematic Business Model-Driven Approach for Deriving UML-Based Requirement Specifications

Abstract

Requirement engineering, or elicitation as also commonly known, is a critical stage in software engineering that enables requirement engineers extract the right knowledge of the system, functional and constraints, needs of stakeholders. This is often both an error prone and a time expensive process. Research shows that the majority of the errors in the software functionality, often discovered in later stages of development, are directly linked to the mistakes (e.g. misunderstanding, ambiguity/interpretations, correctness, etc.) made during the requirement elicitation phases. Therefore, many researchers proposed different approaches to enhance the existing requirements engineering techniques to both reduce such mistakes and speed up the requirements engineering process. One of these approaches is based on utilizing business process modelling to take benefit from business process models that are already available in an enterprise to derive requirements. Many organizations have their existing business process models in the form of working instructions, consisting of embedded business rules and processes that provide enough valid details for specifying, important aspects of, software systems. Upon reviewing the research done in this field, many approaches have been proposed to automatically generate requirements specifications from business models (see chapter 3), but they fail to achieve transformation without significant manual intervention. In this thesis, we propose a structured approach to derive system requirements based on business process models (named BMSpec) that employ a set of systematic steps to improve business process models before transforming them into requirement models. The transformation is then done by mapping objects in a business process model (represented in XML/BPMN) to a UML-based use case model and structured-text use cases descriptions. The transformation is performed by algorithms that base their methods on a set of heuristic rules developed part of this research. Although this transformation is not fully automated, but it aims to achieve a greater degree

of automation with more precise and valid requirement specifications, thus enabling to overcome many of the problems that arise in the requirement elicitation phases, such as misunderstanding of the business by requirement engineers, lack of focus on the objective of the system, and miscommunication between stakeholders and system requirement engineer, and reducing effort, time and eventually errors in the software specifications. 37 business process models, classified into 6 levels of richness, have been used to validate and evaluate BMSpec, in terms of its efficiency, sufficiency and correctness. The evaluation results show that BMSpec's generated requirements were correct for the tested case studies, 90% sufficiency, of covering BPMN's notations, and higher generation efficiency directly proportional to the level of richness of the input business process model.

نهج منظم يعتمد على نماذج الاعمال لإشتقاق متطلبات النظام ممثلة بلغة التصميم الموحدة

ملخص

تعتبر مرحلة هندسة المتطلبات المرحلة الأكثر أهمية في مراحل هندسة البرمجيات، لأنها تمكن المحللين من استنباط متطلبات النظم البرمجية بشكل صحيح وتحديد الاحتياجات والشروط اللازمة لتطويرها، حيث تبين البحوث ان معظم الاخطاء في النظم البرمجية التي يتم اكتشافها في مراحل لاحقة من التطوير، تكون عائدة إلى اخطاء حدثت في مرحلة هندسة المتطلبات. لذلك، اقترح العديد من الباحثين أساليب مختلفة لتحسين هذه المرحلة و تقليل الاخطاء التي تحدث اثنائها، احد هذه الاساليب المقترحة يعتمد على النماذج الادارية المتوفرة في المؤسسات لاشتقاق متطلبات النظام، حيث ان العديد من المؤسسات لديها نماذج ادارية على شكل تعليمات أو دليل اجراءات أو مخطط لتمثيل سير العمل، وتحتوي هذه النماذج على تفاصيل كافية صالحة لتحديد جوانب هامة من متطلبات النظم.

بعد مراجعة الأبحاث التي أجريت في هذا المجال، تبين وجود العديد من المقترحات لاشتقاق متطلبات النظم من نماذج الأعمال (انظر الفصل 3) بشكل اوتوماتيكي، ولكن فشلوا في تحقيق الاشتقاق دون اللجوء إلى تصحيح يدوي كبير. في هذه الأطروحة ، نقدم مقترح منظم لاشتقاق متطلبات النظام استنادا إلى النماذج الادارية، حيث يتكون هذا المقترح من مجموعة خطوات لتحسين النماذج الادارية قبل تحويلها إلى متطلبات النظام.

تم تنفيذ الاشتقاق باستخدام خوارزميات مبنية على مجموعة من القواعد تم بناءها كجزء من هذا العمل. على الرغم من أن هذا الاشتقاق ليس مؤتمت بالكامل، لكنه يهدف إلى تحقيق درجة أكبر من الاتمته لاشتقاق متطلبات أكثر دقة وصحة ومطابقة لاحتياجات العمل، وبالتالي يمكننا من التغلب على العديد من المشاكل التي تظهر في مرحلة هندسة المتطلبات، مثل سوء فهم الأعمال من خلال المحللين، وعدم التركيز على الهدف من النظام، وسوء الفهم بين أصحاب المصلحة والمحللين، واختصار الجهد والوقت، وفي نهاية المطاف اختصار الاخطاء التي تظهر في النظم البرمجية.

تم استخدام 37 نموذج اداري مصنفة إلى 6 مستويات من ثراء المحتوى، لتقييم المقترح والتأكد من صحته، حيث تم تقييم المقترح من ثلاثة جوانب : الكفاءة ، الشمولية والدقة. أظهرت نتائج التقييم أن النهج المقترح أنتج متطلبات صحيحة مقارنة بنتائج الدراسات الاخرى، كما وصلت نسبة الشمولية وتغطية رموز BPMN إلى 90% ، وتتناسب كفاءة المقترح تتناسب طرديا مع مستوى ثراء النماذج الادارية برموز BPMN.

Acknowledgements

I would like to express my sincere gratitude to those who gave me the assistance and support during my master study especially my Husband Mohammad Batta.

I would like to thank all three professors, Dr. Adel Taweel, Dr. Dirar Elayyan, Dr. Mohannad Jabari, who served on my thesis examination committee. Their comments and suggestions were invaluable. My deepest gratitude and appreciation goes to my supervisor Dr. Adel Taweel for his continuous support and advice at all stages of my work.

Thanks are extended to my parent who gave me the hope to continue my work.

Another word of special thanks goes to Birzeit University, especially for all those in the Faculty of Graduate Studies / Computing Program.

Table of Contents

ABSTRACT.....	II
ACKNOWLEDGEMENTS	V
TABLE OF CONTENTS	VI
LIST OF FIGURES	IX
LIST OF TABLES	XI
ABBREVIATIONS.....	XIII
1. INTRODUCTION.....	1
1.1 INTRODUCTION.....	1
1.2 PROBLEM STATEMENT	2
1.3 RESEARCH MOTIVATION	5
1.4 RESEARCH GOALS	8
1.5 RESEARCH METHODOLOGY.....	9
1.6 ORGANIZATION OF THE THESIS	10
2 BACKGROUND.....	12
2.1 REQUIREMENT ENGINEERING TECHNIQUES.....	12
2.1.1 <i>Selected traditional requirement elicitation and analysis techniques</i>	12
2.1.2 <i>Requirement specifications</i>	14
2.2 BUSINESS PROCESS MODELLING	15
2.3 USE CASE MODEL	16
3 LITERATURE REVIEW	18
3.1 INTRODUCTION.....	18
3.2 GOAL-ORIENTED REQUIREMENT ENGINEERING APPROACHES	18
3.3 ORGANIZATIONAL –ORIENTED APPROACHES	20
3.4 TASK ORIENTED APPROACHES	23
3.5 DISCUSSION.....	30
4 PROPOSED APPROACH: BMSPEC.....	33
4.1 INTRODUCTION.....	33
4.2 FIRST STEP: PREPARING A WELL-DEFINED BUSINESS PROCESS MODEL.	34
4.3 SECOND STEP: GENERATING USE CASE DIAGRAM.	36
4.3.1 <i>Generate Items in use case diagram</i>	37
4.3.2 <i>Generate Associations in use case diagrams</i>	38
4.4 ALGORITHM TO GENERATE USE CASE DIAGRAM	42
4.5 THIRD STEP: GENERATING USE CASES DESCRIPTION.....	45
4.5.1 <i>Data Association:</i>	45
4.5.2 <i>Artifacts:</i>	47

4.5.3	<i>Message Flow:</i>	47
4.5.4	<i>Sequence Flow between two activities:</i>	48
4.5.5	<i>Sequence Flow between gateway and activities</i>	49
4.5.6	<i>Sequence Flow between <<Event>> and <<Activity>></i>	50
4.5.7	<i>Sequence Flow between Activities and Events</i>	52
4.6	ALGORITHM TO GENERATE USE CASE DESCRIPTION.....	53
5	EVALUATION	57
5.1	EVALUATION METHODOLOGY.....	57
5.1.1	<i>Business process model selection</i>	58
5.1.2	<i>Efficiency evaluation:</i>	58
5.1.3	<i>Sufficiency Evaluation</i>	62
5.1.4	<i>Correctness Evaluation</i>	63
5.1.5	<i>Evaluation Method</i>	63
5.1.6	<i>Experiment Design</i>	64
6	RESULTS, ANALYSIS AND DISCUSSION	66
6.1	EFFICIENCY EVALUATION RESULTS	66
6.2	CORRECTNESS EVALUATION RESULTS	80
6.3	SUFFICIENCY EVALUATION RESULTS	93
7	CONCLUSION	98
7.1	INTRODUCTION.....	98
7.2	CONTRIBUTION.....	99
7.3	RESULTS.....	100
7.4	LIMITATIONS AND ASSUMPTIONS	102
7.5	FUTURE WORK:.....	102
	REFERENCES	104
I.	APPENDIX: EXAMPLE HOW BMSPEC GENERATES USE CASE MODEL	112
	□ REQUEST FOR PROPOSAL PROCESS	112
	□ CAB BOOKING PROCESS	116
II.	APPENDIX: EXAMPLE OF COMPARISON BETWEEN MANUAL WORK AND BMSPEC RESULTS	120
	EXTRA FEATURE CALCULATOR	120
	REGISTRATION ON X-ROAD	123
	CONSUME X-ROAD SERVICE.....	125
	PROVIDE X-ROAD SERVICE	127
	ONLINE BOOKSHOP SYSTEM(MAKE ORDER PROCESS).....	129
III.	APPENDIX: GOLD STANDARD CASES	131
IV.	APPENDIX: BUSINESS PROCESS MODELLING NOTATION (BPMN)	134

V.	APPENDIX: SOURCE CODE FOR GENERATING USE CASE DIAGRAM	136
VI.	APPENDIX: SOURCE CODE FOR GENERATING USE CASE DESCRIPTION	166

List of Figures

Figure 1:Software system errors [63]	3
Figure 2:main element in use case diagram [22]	15
Figure 3:use case diagram example[25]	17
Figure 4: use case description example[31].....	17
Figure 5: Example of Map diagram [37]	19
Figure 6: “Mapping from business process to use case concepts” [1].....	24
Figure 7: BMSpec.....	34
Figure 8: R1	37
Figure 9:R2	37
Figure 10: R3	38
Figure 11: R4	40
Figure 12: R5	40
Figure 13: R6	41
Figure 14: R7	41
Figure 15: R8	41
Figure 16: R9	42
Figure 17: R10	42
Figure 18: adopted Use case description template [31]	45
Figure 19: the effect of model richness on the efficiency of BMSpec.	71
Figure 20: the effect of model richness on the efficiency of BMSpec.	71
Figure 21: comparison between different levels of richness.	78
Figure 22: The Nobel Prize Use Case Diagram [31].	86
Figure 23: The Nobel Prize Use Case Diagram From BMSpec	86
Figure 24: Send Nomination Form use case description[31]	87
Figure 25: Send List of Preliminary Candidates use case description [31]	87
Figure 26: Write Recommendations use case description[31]	88
Figure 27: Search for car use case description from manual work.....	92
Figure 28: Hire car use case description from manual work	92
Figure 29: sufficiency of proposed approach	97
Figure 30 : usage covered relation between notations	97

Figure 31: Request for Proposal process model.	113
Figure 32: Request for proposal use case diagram.	113
Figure 33: Cab booking BPMN Process model.....	116
Figure 34: Cab booking use case diagram.....	117
Figure 35: Do Simple Calculation use case description from manual work	121
Figure 36: Do Money Exchange use case description from manual work	122
Figure 37: Use case diagram of X-road registration from manual work	123
Figure 38: Use case diagram of X-road registration from BMSpec	123
Figure 39: Fill Form to register X-road use case description from manual work.	124
Figure 40: send confirmation use case description from manual work	124
Figure 41: consume X-road service use case diagram from manual work.....	125
Figure 42: consume X-road service use case diagram from BMSpec	126
Figure 43: Validate consume form use case description from manual work.....	126
Figure 44: Confirm consume form use case description from manual work.....	127
Figure 45: provide X-road service use case diagram from manual work.....	127
Figure 46: provide X-road service use case diagram from BMSpec.	128
Figure 47: Fill provide form use case description from manual work.....	128
Figure 48: Publish provide form use case description from manual work.	129
Figure 49: make order use case diagram from manual work.....	130
Figure 50: Figure 49: make order use case diagram from BMSpec	130
Figure 51: Flow objects [24].....	134
Figure 52: Connecting objects [24]	134
Figure 53: Swimlanes [24].....	135
Figure 54: Artifacts [24]	135

List of Tables

Table 1:Scenario sentence generated by Data Associations [31]	27
Table 2:Pre-condition sentences generated by gateways [31]	27
Table 3:Triggers generated by events [31]	28
Table 4:Analysis of task oriented approaches for deriving use case model.	31
Table 5: Rules to generate use case diagram	38
Table 6: Rules to generate use case associations	39
Table 7: Rules to transform data association	46
Table 8: The sentence generated by text annotation.	47
Table 9: sentence generated by message flow.	48
Table 10: sentence generated by sequence flow between two activities.	48
Table 11: The sentence generated by sequence flow between gateways and activities	49
Table 12: The sentence generated by sequence flow between events and activities ...	51
Table 13: The sentence generated by sequence flow between activities and events. ...	52
Table 14: LOR1 OF Richness.....	59
Table 15: LOR2 OF Richness.....	59
Table 16: LOR3 OF Richness.....	60
Table 17: LOR4 OF Richness.....	60
Table 18: LOR5 OF Richness.....	61
Table 19: LOR6 OF Richness.....	62
Table 20: notation in open vacancy model	67
Table 21: The effect of model richness on the efficiency of BMSpec.	68
Table 22: result of running the proposed algorithm on thirty business process model classified in to six level of efficiency	73
Table 23: Effect of number of input notation on the efficiency of BMSpec.	74
Table 24: time and type of model transformation.....	79
Table 25: time and type of business process model transformation using BMSpec. ..	80
Table 26: time and type of business process model transformation using manual work.	81

Table 27: comparison between time required for manual work and BMSpec	82
Table 28: result of running the proposed algorithm on seven business process model.	83
Table 29: comparison between result of BMSpec and manual work with correctness ratio	85
Table 30: Send Nomination Form use case description From BMSpec	87
Table 31: Send List of Preliminary Candidates use case description from BMSpec ..	88
Table 32: Write Recommendations use case description From BMSpec	88
Table 33: comparission between use case diagram generated from manual work and BMSpec	91
Table 34: Search for car use case description from BMSpec	92
Table 35: Hire car use case description from BMSpec	93
Table 36: the result of sufficiency evaluation: covered and uncovered notation	95
Table 37: How BMSpec addresses limitaions in related existing work.	99
Table 38: Simple Calculation use case description from BMSpec	121
Table 39: Do Money Exchange use case description from BMSpec	122
Table 40: Fill Form to register X-road use case description from BMSpec	124
Table 41: send confirmation use case description from BMSpec.	125
Table 42: Validate consume form use case description from BMSpec	126
Table 43: Confirm consume form use case description from BMSpec	127
Table 44: Fill provide form use case description from BMSpec	128
Table 45: Publish provide form use case description from BMSpec.	129

Abbreviations

BPMN	Business process modelling notation
UML	Unified modelling language
XML	Extensible Markup Language
BPEL	Business Process Execution Language
RAD	Role Activity Diagrams
Bizagi	Business Process modeller
XPDL	XML Processing Description Language
GM	Goals Model
EKD	Enterprise Knowledge Development
BRM	Business Rule Model
CM	Concepts Model
BPM	Business Processes Model
ARM	Actors and Resources Model
TCRM	Technical components and requirements model
MDA	Model driven Architecture
BPR	Business Processes re-engineering
LOR	Level Of Richness
BMSpec	Business Model derived Spec

Chapter 1

Introduction

This chapter introduces the thesis. It describes the problem statement, motivations, goal and objectives, and organization of the thesis.

1.1 Introduction

Requirement engineering is a critical stage in software development [27]. Employing effective requirements engineering techniques and methods are essential to the success of software development projects, not only for achieving them on time and within budget but also for delivering the desired business value [28]. Research has shown that many large projects fail because of inadequate requirements, showing that errors made in the requirements engineering stage “are among the most difficult to detect and the most expensive to correct” [10]. Many approaches have been proposed to enhance the existing requirements engineering techniques, some of these approaches recognized that “understanding a business process is the key to identify the user needs of the software that supports it” [29, 30, 31]. Many organizations have their existing business process models in the form of working instructions and include enough valid details for specifying software systems, which thus may provide a basis for understanding and modelling software requirements. Business process based approaches, in requirement engineering, can enhance the quality of software requirement because it ensures better understanding and validity of business models. On reviewing the research done in this field, we found many procedures and algorithms that have been proposed to automatically generate requirements specifications from business models, but these approaches fail to achieve transformation without

significant manual intervention or correction. In this thesis, we propose a new systematic business model-driven approach for deriving UML-based requirement specifications (named BMSpec). It employs a set of systematic steps that start by improving the existing business process model to result into a well-defined business model, which includes the effect of the prospective “information system should have on the business processes” (named “To-Be” model) [29]. The second step automatically transforms the “To-Be” business process model to a Use Case Model, inclusive of both the Use Case diagram and the detailed Use Case descriptions. The main aim of BMSpec is to automate the generation of valid requirement specifications that meet the business needs as represented in and derived out of the business model, thus overcoming the often used tedious and time-consuming manual process. Upon evaluation, BMSpec shows accurate results when compared with other manual approaches, it also shows higher generation efficiency directly proportional to the level of richness of the input business process model. To the best of our knowledge, it is the first approach that can automatically transform business process models to Use Case models that include generated detailed use case descriptions and associations between use cases within minimal manual intervention.

1.2 Problem Statement

Requirement engineering is a critical stage in software development, it is the initial step and it “can affect the entire software development activity if not properly done” [62]. The deciding what to build is the hardest part of building software system, No other part of the work can more damage the resulting system if done wrong. No other part is more difficult to be corrected later. At the same time, it can be “a key to the successful completion of software projects” [70]. “In large and complex systems, it is difficult to develop accurate requirements” [70]. Researches show that “errors in requirement elicitation phase are the most difficult and most expensive in software development” [63]. Studies by [63], indicate that “70% of the system errors relate to inadequate system specifications”

and “30% of the system errors are due to design issues” [63], as illustrated in Figure 1

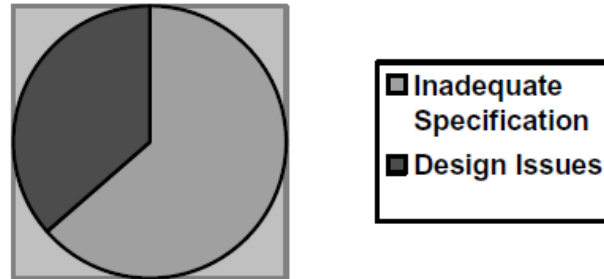


Figure 1: Software system errors [63]

Many problems appear in the requirement elicitation phase that may cause these errors. [64] Classifies the common elicitation problems into three main types:

- **Problems of scope:** the boundary of any system is not completely clear or determined, which may result into unnecessary information be included and necessary information may be excluded from requirements. For more accurate requirements, the boundary of the target system and the objectives of the system must be defined. The boundary of the target system can be determined by starting the requirements elicitation with organizational and contextual analysis. It must focus on users' needs and concerns and not just developers' needs.
- **Problems of understanding:** such problems can result into an incomplete, ambiguous, inconsistent and also incorrect requirements. When the analyst and developer fail to determine the important questions that address the stakeholders' true needs or when the stakeholders are not completely know their needs, which consequently will raise several issues due to the lack of user input. [65] found that 56% of such errors, which were due to poor communication between users and analysts". These errors were the "most expensive to correct taking up to 82% of staff time" [65].

The analysts may have poor knowledge of the system domain and sometimes the domain experts and analysts speak different languages, or different users

may have conflicting needs. When stakeholders and analyst speaks different languages, the language becomes a big barrier. But also when they speak the same language, “stakeholders from different domains (e.g. manufacturing, technical, marketing and management) use the same words and terms with different meaning” [65].

- **Problems of volatility:** requirements grow and change over time because of changing needs and changing interests and goals of stakeholders. “The major cause for requirements volatility is that users’ needs evolve over time” [65].

Volatility can arise because the requirement is the product of participation of many stakeholders, and these stakeholders have conflicting objectives, needs views and interests [69]. Also, it can arise because clients or customers usually don’t completely understand the capabilities and limitations of the technology.

They may have unrealistic expectations from the software system.

Several research groups proposed methodologies with processes, models and techniques in requirement engineering such as goal-oriented approaches, organizational-oriented approaches, and business process based approaches [36, 37, 39, 44].

But not all of these approaches can produce an accurate, efficient, complete and on time requirements, In this thesis, BMSpec bases itself on the underlying business process for several reasons: First, many researchers “have recognized that understanding a business process is the key to identify, analyse and validate requirements for the information system to be developed” [71, 72]. Second, business process modelling is an approach to graphically display the way organizations conduct their business processes and represent an organizational documentation, thus a business model helps to determine the organization structure and the goal of the system [49], which is a very important factor in order to prevent the “problem of scope” that can arise in the requirement elicitation phase. Third, the business process model is the basis that allows different people to communicate and understand the running of the business in an organization. When a business process model is represented in PBMN is thus “readily understandable by majority of, if not all, business users, from the business

analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes” [72], this is a very important factor which can avoid the “problem of understanding” that can arise in the requirement elicitation phase. But how to design a well-defined business process model that can be automatically transformed to requirement specifications can be extremely challenging. Several approaches have been proposed to generate requirement specifications from business process models, but most of them either use a public rule or a general pattern with manual transformation. Approaches that employ some automated transformation do not cover relations or associations between Use Cases, nor generate the detailed use case descriptions.

This thesis aims to develop new approach to automatically derive requirement specifications from business process models to produce more accurate and valid requirements. It also aims to reduce the effort and time of the requirement engineering process by automating the generation of, as many as possible, of the Use Case model elements from the business process model in an enterprise, compared to building it from scratch using traditional manual techniques.

1.3 Research Motivation

- **Why requirement engineering is important?**

Research has shown that many large projects fail because of inadequate requirements [64]. At the same time errors made in the requirements engineering stage ” are among the most difficult to detect and the most expensive to correct “[10]. Requirement engineering considered the most difficult and critical stage in software engineering development cycle [10]. Many approaches have been proposed to produce more robust requirement specifications and speed up the process of requirement engineering [10, 64, 65].

- **Challenges of requirement elicitation?**

Many problems may arise in the requirement engineering phase as we mentioned in section 1.2. These problems may lead to an unsatisfactory or unacceptable

developed software system, or a system accepted, but with either high maintenance cost, or endless frequent changes.

The requirement engineering process can be divided into three activities [52, 53]: requirement elicitation, specification and validation. By improving requirement elicitation and automatically generate the requirement specification, the requirement engineering process can be improved, errors can be reduced, which result into an enhanced requirements and a more robust, accurate, complete system.

In this thesis, we propose a new approach to improve requirement elicitation using business process modelling, business process reengineering and a rule-based transformation algorithm that automatically generates requirement specification into a use case model.

- **Why business process modelling?**

In recent years, requirements of business applications have changed from command-based applications to workflow-based applications, “at least half of industrial software development is connected to business application development” [71]. So business process modelling is a suitable approach to help in producing accurate requirements specification:

- Business process modelling solves the “problem of understanding”, which may arise in the requirement elicitation phase by creating comprehensible models with clear notation. These models, represented in PBMN, “are readily understandable by majority of, if not all, business users, including business analysts that create the initial drafts of the processes, to technical developers responsible for implementing the technology that will perform those processes” [21].
- Business process modelling allows requirements engineers to understand the business environment through “as-it-is” and “To_be” modelling. “Both stakeholders and developers can negotiate on which system is to be built and what process is manually achieved or automatically achieved by the system” [72].

- Business process modelling solves the “problem of scope” that can arise in the requirement elicitation phase by displaying the way organizations conduct their business processes and represent an organizational documentation, thus it helps to determine the organizational structure and the goal of the system [49].

- **Limitations of Existing Work**

Business process-based approaches in requirement engineering are efficient approaches, but most of them suffer from several limitations. These include, weak coverage, complex to implement, redundant result, need manual work and incomplete output requirement specifications. Some of these approaches [1, 2, 6] attempt to transform automatically business process models, represented in a UML activity model, to a UML use case diagram, but they transform all types of processes (system and manual processes) - they assume that all business processes are supposed to be computerized. There are often, in a business, some processes that are more suitable for being performed by hand and cannot be achieved by software systems, which leads to creating redundant and complex use cases. In addition, these approaches do not handle include, extend, or generalization relations between use cases. So, the resultant use case model needs manual reconstruction. Another limitation of these approaches is that their focus was on generating the use case diagram only without the description.

Other approaches propose a manual transformation from business model, represented in BPMN, to requirement specification represented in a use case model [31, 7, 9, 3]. These approaches use a set of rules, to build a well-defined business process model, which includes automated processes. They address some of the limitations that appear in the automated approaches such as determining automated processes and association between generated use cases. However, they use a manual approach, i.e. the transformation is done manually, which can take significant time and high probabilities of arising errors. One of these approaches focuses on generating use case descriptions more than use case diagram details

[31]. It generates use case descriptions from a specified set of predefined natural language sentences mapped from BPMN model elements.

Most of reviewed approaches [1, 2, 3, 6, 9, 31] don't handle use case descriptions and neglect associations between use cases such as inclusion, extension or generalizations of use cases, so obtained use case diagrams need manual correction.

1.4 Research Goals

The overall goal of this thesis is to improve the requirement engineering process and produce a more accurate and more related to business needs requirement specifications through automating the generation of as many as possible of the elements of the use-case model, thus saving significant effort and time.

To achieve this overall goal, the following research objectives have been defined:

- Create a set of rules to achieve Business process reengineering, in order to build a well-defined business process model. We use business process reengineering techniques to determine the effect the information system should have on business processes. In this step, we analyse the purpose and goal of the system. In order to reengineer the business process model effectively, we should focus on both the system perspective (i.e. defining a well-defined use case) and business process perspective (i.e. defining what is needed from the Information System) [20]. In BMSpec, we should identify the tasks that represent interactions with the software system. Because only these tasks are important to generate use case model. Any well-defined use case must specify a functionality, which an actor wants to achieve by using the system.
- Develop a set of rules to build well-defined business model, in order to get an accurate and precise use case model. For example, ensure complete definitions of gateways, triggers, actors as roles, assign appropriate (performer) for each user task, etc. avoid using pools and lanes to represent participants, enrich model with all required data objects as input or output and specify the fields of data objects, and so forth.

- Create a set of heuristic rules to map business process concepts to use case diagram concepts
- Develop a transformation algorithm, which automatically transform business process models to UML use case models. Based on a predefined set of heuristic rules, the algorithm creates a mapping between the two meta-models by evaluating their definitions, maps concepts or relations in the business process meta-model to concepts or relations in the use case meta-model.
- Create a set of heuristic rules to map business process concepts to use case description details.
- Develop another algorithm to automatically transform business process model to a detailed description of use cases.
- Develop an evaluation methodology and evaluate the performance of BMSpec and developed algorithm. The developed evaluation methodology includes three types of evaluation: model efficiency, sufficiency, and correctness. 37 different business models have used to evaluate BMSpec.

1.5 Research Methodology

This section describes the research methodology that was followed:

- 1- Carry out a detailed literature review of existing relevant approaches in requirement engineering. Three main types of modelling approaches have been found: organizational based, goal- based and business based approaches. We focused on business process based approaches and identify limitations of these approaches. We find that these business process based approaches in requirement engineering can be categorized into: automated transformation from business process model to UML use case diagram, and manual transformation using general pattern and reengineering techniques to manually extract use case model.
- 2- Collect and study different types of different business models to identify the potential levels of richness and their representation. In addition, study the Business Process Modelling Notation (BPMN) to identify the potential

coherency of the notation and its semantic consistency for model representation and its potential for requirement specification derivation.

- 3- Develop and propose a new systematic approach that automatically generates use case models derived from business process models. BMSpec follows three systematic steps: First, prepare a well-defined business model, this step depends on the status of the existing business process model it is a manual step. Second, automated transformation from business process model to UML use case model, in this step we write an algorithm to map XML objects in both models, a set of heuristic rule are used to transform objects and relations from business process model to use case diagram. Third, automated transformation from business process model to use case description by using a set of heuristic rules, in this step we write an algorithm to map notations from business process model to a detailed use case description.
- 4- Develop an evaluation methodology and evaluate the approach. The evaluation methodology evaluates the approach on its sufficiency, efficiency and correctness. The evaluation was conducted in two stages: conduct dry run experiments based on the evaluation methodology by running developed approach arbitrary three business process models to check if the evaluation methodology generates the sought after data; conduct wet run experiments based on the evaluation methodology by running developed approach on 37 business process models to evaluate the correctness, sufficiency and efficiency of the approach.
- 5- Analyse experiments' data and results: to calculate the sufficiency of the approach in representing BPMN semantic representations; the effect of the level of richness of the input business model on the efficiency of the approach; and the correctness of BMSpec by comparing its results with the results of other approaches.

1.6 Organization of the Thesis

The remainder of this thesis is structured as follows:

Chapter 2: Background. Presents the basic concepts of business process modelling, business process re-engineering, traditional requirement elicitation and requirement specification techniques.

Chapter 3: Literature Review. Reviews related works in specifying requirement based on business process and focus on task oriented business process modelling.

Chapter 4: The proposed approach (BMSpec). Proposes a new systematic approach that generates use case model with use case description dynamically based on business process model.

Chapter 5: Evaluation. Examines the correctness, sufficiency and efficiency of the proposed approach. We use 37 business process model in the evaluation.

Chapter 6: Results, Analysis and Discussion. Analyse and discuss results of BMSpec with respect to three key characteristics, efficiency, sufficiency and correctness.

Chapter 7: Conclusion. It discusses the conclusions of thesis, limitations and suggests some possible future work.

Chapter 2

Background

This chapter aims to provide a general discussion of the concepts needed to understand the rest of the thesis. It covers basic concepts of traditional requirement elicitation techniques, requirement specifications, business process modelling, and use case model.

2.1 Requirement engineering techniques

The requirement engineering process starts with the requirements elicitation. The system analysts and requirements engineers collect information from the domain experts. Information can be gathered from documents, legacy applications, interviews, etc. which are used in the specifying of the requirements in this activity called requirements specification, and the requirements validation used to find out if there are some errors or undefined requirements.

In the following subsection we will explain the meaning of requirement elicitation and present four techniques used to gather requirements, requirements specification and present two technique used to specify requirements.

2.1.1 Selected traditional requirement elicitation and analysis techniques

Requirement elicitation techniques used to extract the right knowledge form the stakeholders. Despite the importance of the requirement elicitation process, little research has focused on most suitable elicitation techniques [52]. There are many elicitation technique published in papers and research but there is no common agreement on how to select one or combined techniques together to work efficiently in a specific situation. Below describes 4 techniques, many other techniques and details can be found in [77].

2.1.1.1 Interviews

Interviews are considered one of the traditional and most often used techniques [53]. In this technique, the analysts and requirements engineers make a discussion with different type of stake holders to get information about the software system. According to [53] there are two main types of interview:

- **Closed interview:** In closed interview requirement engineer define list of questions before the interview, and during the interview they try to get answers of these questions from the domain experts and stakeholders.
- **Open interview:** In open interviews, requirement engineers try to make an open discussion with stakeholders to get information about the system especially the stakeholder's expectation from the system.

2.1.1.2 Questionnaire

Questionnaire is “a technique that consists of preparing documents with a list of questions filled by stakeholders” [54]; the answers of these questions must be short so as stakeholders will not feel bored, so the answers can be prepared as checkbox. But there are many factors which affect usage of questionnaire [54]: the available resources to gather requirements and type of requirements to be gathered some domain problems may be not known to the requirements engineer.

2.1.1.3 Brainstorming

Brainstorming is a technique often used to solve a problem and generate new ideas [56]. It is a kind of mini conference, held among six to ten experts “one of them has to assume the role of a moderator, but should not control the session” [55]. It is usually held in a round table fashion where every member has period of time to explain his ideas. The team of brainstorming decides the best idea as a solution to the issue discussed in the conference, this discussion done by voting [57].

2.1.1.4 JAD (Joint Application Development)

“It is an organized and structured technique used in requirements elicitation” [58], similar to brainstorming, except the stakeholders allowed to participate and discuss the design of proposed system. The total number of participants do not exceed 30 people [58]. The requirement engineer starts the session by displaying general description of the proposed system, then the discussion with stakeholders continues until final requirements gathered [57].

2.1.2 Requirement specifications

After gathering the requirements it should be documented in precise detail to be understandable by the stakeholders and system developers. The requirements can be documented by both lists of textual requirements usually written in natural language and graphical models or can be brought together by combining the two approaches. There are several requirements specification techniques we will present the most popular two techniques, many other techniques and details can be found in [77].

2.1.2.1 Scenarios

“Scenarios consist of the description of the characteristic of the application by means of a sequence of steps, it describes a system from a user’s perspective, focusing on user-system interaction” [59]. It can be represented in different ways: text, structured text, images, animation or simulations, charts, maps, etc. According to [60]: Scenarios technique has many advantages: First, “A scenario views a system from the viewpoint of one of its users”. They give the users a feel for what they will get. Second, “the strength of scenarios lies in the fact that they divide the system into functions from a user’s perspective and that each function can be treated separately”. Third, short feedback cycles between stakeholder and requirements engineers because scenarios use user- oriented way of representing requirement and each function can be treated separately.

2.1.2.2 Use case modelling

“A use case is a description of set of sequence of actions that system performs” [25]. It is used to structure the behavioural aspects in a model. “Use case model is designed for software or system designer, not for any business people. There are three main elements in a use case diagram: use case, actor and association link to connect actor with use cases each element” [25] is defined in Figure 2.



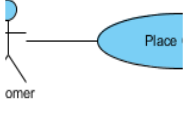
Notation	Description
	Use case - Each use case represents a user goal, which is an objective the user of the system wants to achieve. Note that use cases can only be used to show what the user wants to do instead of what the developer needs to develop, although they may be the same in some cases. If you want to document or model the functions involved in a use case, you may use the flow of events tool, or to elaborate a use case with sequence diagram/activity diagram . Just keep in mind that use case modeling aims at modeling what the user wants to achieve.
	Actor - User of the system. The word 'user' here is not limited to humans. It can be a system that interacts with our system to fulfill certain business objective.
	Communication link/Association - Connects between actor and use case to indicate the access of system by actor. Each communication link implies a sequence of transactions between actor and system.

Figure 2:main element in use case diagram [22]

2.2 Business process modelling

Business process modelling has become a very common practice in organizations, it enables a common understanding analysis of a business process, and thus helps in achieving goals, and increasing competency in organizations [49]. “Business process model describes graphically at least the activities, events and control flow that specify business processes” [49]. The model May also include information about the involved data, organizational and IT resources [50].

Business process models allow people to communicate and understand the running of an organization. They are used for multiple purposes, such as business process management, software development and business process reengineering.

According to [51], process model provides a comprehensive understanding of a process, helps organizations in decision about process development and design, facilities control and support during process execution and uses for analysis of information technology.

There are many existing notations for business process modelling, each notation has its strengths and weaknesses such as UML-activity diagram [61] and BPMN [24]. In BMSpec, we use the Business Process Modelling Notation (BPMN), a graphical notation language widely used for modelling business processes. “Developed by Business Process Management Initiative and currently maintained by the Object Management Group” [24]. The main goal of BPMN is “to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes”[21].

2.3 Use case model

The Use Case Model [25, 78] describes the functionality of a software system. “A Use Case represents a unit of interaction between a user and the system”, each Use Case has a description, which describes the functionality that will be implemented in the software system [25]. Use Cases are related to 'actors' who run the functionality. “An actor is a human or a machine entity that interacts with the system to perform meaningful work” [78]. Also a Use Case may 'include' another Use Case's functionality, 'extends' another Use Case, 'invoke' or 'precede' other Use Cases in an execution order. So a typical use case model consists of two main elements:

- A use case diagram, which represents use cases, actors, and associations between them, figure 3 illustrates an example of a use case diagram:

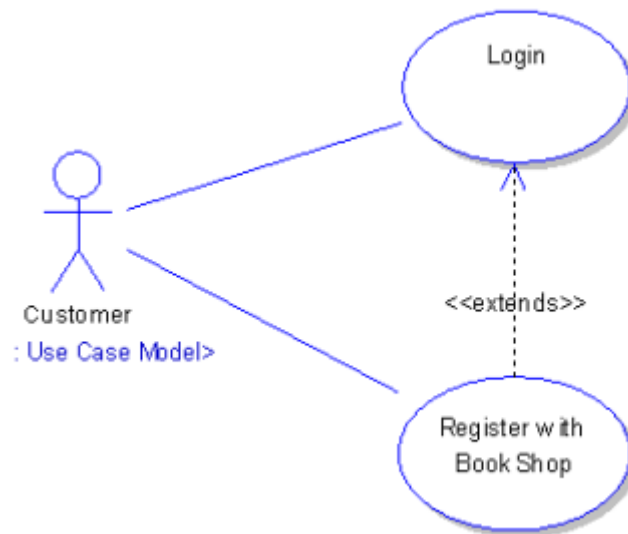


Figure 3:use case diagram example[25]

- A use case text description, which include a detailed description of a use case, defining its general comments and notes, pre-conditions that must be true before the use case is run, post-conditions that must be true once the use case is run, scenarios (or workflow) that defines sequential descriptions of the steps taken to carry out the use case, actors who run the use case and triggers that makes use case execute. Figure 4 illustrates an example of a use case description:

Use Case name	Send Nomination Form.
Actors	Nobel Committee, Nominators
Trigger	The time-date September is reached.
Scenario	Around 3000 invitations confidential nomination forms are sent to selected Nominators. Reads information from Nominators. Sends the Nomination Invitation to Nominators.

Figure 4: use case description example[31]

In this thesis the Use Case Model, include both the Use Case diagram and the detailed Use Case descriptions.

Chapter 3

Literature Review

This chapter aims to review related work in generating requirement specifications based on business process model. We classify the related research into three main types of approaches: Goal-oriented approaches, organizational oriented approaches and task oriented approaches.

3.1 Introduction

Many approaches try to solve problems which may arise in requirement engineering phase, In this literature review, we present and discussed approximately twenty related researches classified into three main types of approaches: first, approaches focus on solving the Problems of scope, such as Goal-oriented approaches; second, approaches focus on solving the problems of understanding such as organizational oriented approaches; third, approaches focus on solving many problems such as task oriented approaches. Where task oriented approaches are used in this thesis, so we will focus on it more than the other two approaches.

3.2 Goal-oriented requirement engineering approaches

“Goals represent the objectives that a software system should achieve in order to meet stakeholders’ needs, they have been recognized to be an essential component of the Requirement engineering process” [35]. Goal-oriented approaches tried to overcome a major drawback of traditional requirement engineering approaches, traditional requirement engineering approaches have focused on the functionality of a system and its interactions with users. Instead of determining what the system needs to do, goal-oriented requirement engineering approaches ask why a given functionality is necessary and how it could be implemented. According to [36] goal modelling and analysis in the requirement engineering process aims to: facilitate requirements elicitation, identify and evaluate alternative

implementations, detect irrelevant requirements, obtain complete requirements specifications, identify and resolve requirements conflicts and define stable goals. Many approaches use Goal- oriented such as: Map “it is a goal/strategy-driven approach to capture the goals of an enterprise or system and determine the strategies that can contribute to the achievement of these goals” [37], Figure 5 presents an example on it. The focus on the concept of strategy as a way to achieve a goal distinguishes Map from other goal-oriented requirement engineering approaches. They focus on the strategies because many stakeholders do not distinction between goals and strategies. “Map diagrams consist of a graph whose nodes are goals and whose edges are strategies. An edge entering a node represents a strategy used to achieve the goal of the node”. Another research which uses MAP approach is [4]. In [4] they propose a business process based requirements analysis that tries to prevent common mistakes such as “the lack of understanding of the business by system analysts, the lack of focus on system purpose, and miscommunication between business people and system analysts”. Their approach combines BPMN and MAP. “BPMN is used for modelling the business processes, and MAP is used for modelling the goals and strategies that lead to the users’ requirements. Initial BPMN models (As-Is) are updated by the results of the analysis of the MAP model to get the To-Be business process models” [4]. They involve the end user in validating (As-Is) and (To-Be) models.

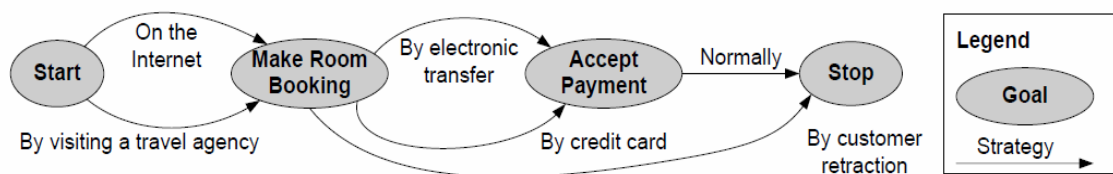


Figure 5: Example of Map diagram [37]

Another approach called the “i* Framework” [41] that uses the goal- oriented, but it focuses on modelling of the dependencies that exist between the business actors in order to achieve organizational goals. The framework consists of two models: first, the strategic dependency model, which describe dependencies that exist

between actors to achieve goals, to perform tasks and to provide or request resources. Second, the strategic rationale model that describes the reasons that exist behind each dependency relationship. This is useful for representing tasks that have to be performed by the actors to achieve their goals. This model is based on the elements of the dependency model, but it also adds task decomposition links (to represent the combination of the important tasks to achieve a goal) and mean-ends links (to present the diverse options that can be taken to achieve a task or goal).

In this section we represent two main examples of goal oriented approaches: MAP [37] and i* framework [41]. But the MAP approach uses two different notations (goals and strategies), which may result in a sophisticated and crowded model that is difficult to link with the next phases of software development process. The business process model in MAP consists of: nodes, which represent activities, and edges, which represents the way and sequences to execute activities. The other example on goal oriented approaches is i* framework, although it declares the tasks and actors and the dependency between them, but it has several weaknesses, according to [42,43], including: the i* diagram might be too complex and difficult to be understood by all stakeholders, it has no enough support aspects such as scalability and refinement. We conclude that goal oriented requirement approaches succeed in solving the problem of scope, which arises in the requirements engineering phase, but the resultant models are crowded and difficult to understand by all participants and stakeholders, which can result in a problem of understanding. Thus we did not adopt the goal oriented approach in our thesis.

3.3 Organizational –oriented approaches

Organizational modelling [38] is the set of activities that are used to develop the various parts of an organizational model. It usually consists of different sub-models for representation of the faces (activity, information, and constraints). The main aim of organizational modelling is to represent and understand the organization for which a software system is going to be developed [39]. Most of

the existing organization modelling-based requirement engineering approaches focus on creation of business process models and elicit system requirements from them [76, 37, 40, 44], one of popular approaches which use organizational-oriented is Enterprise Knowledge Development (EKD) “it provides a systematic and controlled way of analysing, understanding, developing and documenting an organization and its components” [40], the main aim of EKD is to “provide a clear picture of how an organization operates at a given moment, what are the requirements and the reasons for change, what alternatives could be made to meet these requirements, and what are the arguments for evaluating these alternatives” [40]. It consists of six models [40]: first, “Goals Model which focuses on describing the goal of the enterprise”. Second, “Business Rule Model which usually used to define formulated business rules and its consistency with the Goals Model”. Third, “Concepts Model which usually used to define the "things" and "phenomena". It includes organizational concepts, attributes, and relationships”. Fourth, “Business Processes Model usually used to define enterprise processes, the way they interact and the way they handle information as well as material”. Fifth, “Actors and Resources Model which usually used to describe how different actors and resources are related to each other and how they are related to components of the Goals Model, and to components of the Business Processes Model”. Sixth, “Technical components and requirements model, which defines requirements for the development of an information system”.

A different Organizational-oriented approach is Communication Analysis [44] in which business process modelled from a communication perspective and not from a behavioural perspective. It focuses on communicative interactions that occur between an information system (IS) and its environment. In this approach, information systems are considered from different perspectives: software, organizational and social perspective. It proposes the requirements structure that result in an ISs description in five levels [44]. First, System/subsystems level, which represents the first requirement level and refers to an overall description of an organization and its environment; it may involve decomposition of the problem to reduce its complexity when the organization is complex. Second, Process level,

which represents business process descriptions both from a dynamic perspective (flows of communicative interactions) and a static perspective (business objects, glossary). Third, Communicative interaction level, which represents a detailed Description of communication interactive, it contains information about messages and all business objects such as processes and actors. Fourth, Usage environment level, which represents capturing the requirements related to the usage of a software-based IS, the design of the user interfaces and the modelling of object classes that will be stored in the IS memory. Fifth, Operational environment level, which represents the design and implementation of the software components and architecture of a software-based IS. An example of using Communication Analysis approach to derive requirement specification is [46], they Propose a Systematic derivation of class diagrams from communication-oriented business process models, they use business process specifications which include message structures, then processing them in order to obtain class diagram views, which are integrated to result in a UML Class Diagram. Their approach “starts by sorting the communicative events and processing them in order”. A class diagram view can derive from each communication event. This step can be “achieved by processing the message structure that corresponds to the event”. But their approach focuses only on generating the UML class diagram from the class diagram views and does not access the system requirement models such as use case model or the functional requirement. The aim of this approach is to generate source code from the class diagram and not to determine the requirements specifications.

In this section, we represent two main examples of organizational oriented approaches: EKD and Communication Analysis, EKD has rich models which can enable derivation of system requirements, but the effect of system purpose on business processes not strongly addressed, also the business process model does not have relation between actors and process. While Communication Analysis improves specifications of system requirements by addressing quality requirements. But similar to EKD it does not focus on system purpose and its effect on the business process of an organization. So Despite the Organizational -

oriented approaches are a strong approach to represent most parts of an organizational model, and It can solve the problem of understanding it has weakness in specifying the effect of purpose of the information system on the business process of an organization which means it does not solve the problem of scope. So we don't adapt the organizational oriented approach in our thesis.

3.4 Task Oriented approaches

Task- oriented business process modelling precisely defined user needs, it focuses on users doing action with the system, there are many existing approaches which use Task- oriented business process modelling to derive system requirements specifications: We can classify these approaches into two main classifications:

- **Automated approaches to transform business models to requirement specifications.**

[1] proposes an automated approach and an algorithm to transform business models to functional requirements in terms of a use case model. The main objective for this approach is to automate drawing up functional requirements from the existing business model instead of building a use case diagram depending on interviews. This algorithm works by creating meta-models for both the use case diagram and the business process model, then compares definitions in the two meta-models, “which concepts or relations in business process meta-model map to which concepts or relations in the use case meta-model”. Each ‘Step’ concept is mapped to a ‘Use Case’ concept, where a step represents “a sequence of tasks that can be performed by the same role without interruption”. Also, each ‘Role’ is mapped to an ‘Actor’, and “the association between a step and a role to an association between the corresponding actor and use case” as shown in [Figure 6](#).

Business Process Concept	Use Case Concept
Role	Actor
Step	Use Case
Association between Role and Step	Association between Actor and Use Case
Task	Interaction
Task in a Step	Interaction in a Use Case
Transition between Tasks in the same Step	Ordering between Interactions in the same Use Case
Guard on Transition	Constraint on Interaction
Alternative Path through a Branch	Alternative Path Description of a Use Case, or Extending Use Case

Figure 6: “Mapping from business process to use case concepts” [1].

They evaluate the quality of their approach by comparing their results with use cases constructed by performing interviews. They used 6 business processes, from the mortgage department of a bank, in the evaluation and generated 42 use cases. 12 use cases only were found valid; the total error percentage in the generated use case diagram was reported as 40% [1].

Another similar study [2] proposed a method to explore association between the use case model and business model, but they used “Role Activity Diagrams (RADs) to model business processes”. The main function of this method is deriving the use case model from Role Activity diagrams. They use two definitions to derive use cases from RAD [2]:

- I. “Each use-case should embody more than one transaction – otherwise there is no beneficiary of the activities that the information supports”.
- II. “Each use-case should support at least one activity leading to a change of state – otherwise the beneficiary would not receive any information”.

They faced “difficulties in the derivation of use cases from process models”, including the notion of an actor is not clear enough in Role Activity models so “there is no simple mapping of Roles in process models on to Actors in use case diagrams” [2]. They use selection, enrolment and registration process for new

student as an evaluation case. But the results show the transformation cannot achieve well a formed use case model because the RAD notation is incompatible with UML, especially that UML actor is not clearly defined in an Role Activity diagram.

But this work is continued by [6], they proposed a method for deriving system models based on business process models. They suggested that the correspondence between the central notion of 'automated activity' in improved RAD model and that of "action or function" in the use case diagram facilitates the derivation of system models based on business process models. Their method consists of four steps: "develop a business process model using RAD model, identify automated activities, link each business objective with automated activities, and develop use case model based on objectives and automated activities". They used a process of student registration as an example to demonstrate their developed approach and showing its ease of use.

Another model-driven approach that may intersect with our work in generating the use case model from the business process model is proposed by [32], but it has different objectives. It uses Model Driven Architecture to show how to transform from Computation Independent Model, which describes the business model, to Platform Independent Model, which can be represented by UML models. It performs the transformation by creating a good Computation Independent Model level through well-defined rules to achieve rich models that contain required information to facilitate the process of the transformation to the Platform Independent Model level. They represent Platform Independent Model level through various UML diagrams. Their approach follows the Model Driven Architecture approach by representing the business dimension in the Computation Independent Model level through the use of BPMN, and the use of UML in Platform Independent Model. This approach achieves transformation between BPMN and use case model but the transformation target is only for moving between levels of Model Driven Architecture to generate the software code as a final step, not the requirements specification as we are planning to do. So the result use case model of their approach will not contain all the details needed for

requirement specifications. They use Query/View/Transformation (QVT) to achieve the transformation, which is related to model driven architecture.

- **Manual approaches to transform business model to requirement transformation.**

A manual transformation approach is proposed, by [31], to obtain use case model based on business process model. It is the first approach that tries to generate a use case description from business process model. So they focus on generating the use case description more than the use case diagram. The use case diagram is generated manually using the following rules:

R1: “A role played by a participant must be represented by an actor in the use case diagram”.

R2: “A lane can be the sub-division of a pool or a sub-division of another lane. Subdivisions represent the actors' hierarchy”

R3: “Each activity will be represented as a use case in the use case diagram”.

R4: “An actor that represents a pool (or a lane) is related with all use cases representing the activities that belong to the pool (or lane)”.

R5: “The actor that represents the participant that sends (or receives) a message to an activity is related to the use case that represents that activity.”

The result use case diagram is not detailed enough because they don't cover the association between use case such as extend, include and generalization.

The use case descriptions are specified from a set of predefined natural language sentences mapped manually from BPMN model elements as follows:

Data association is transformed to sentence in scenario use case description as shown in Table 1.

Data	Graphical representation	Originated sentence in use case scenario.
Data Object as data association source		Receives <data object name>.
Data Object as data association target		Sends <data object name>.
Data Input		Receives <data object name>.
Data Input Collection (Input set)		Receives a collection of <data object name>.
Data Output		Sends <data object name>.
Data Output Collection (Output set)		Sends a collection of <data object name>.
Data Store as data association source		Reads information from <data store name>
Data Store as data association target		Writes information on <data store name>

Table 1:Scenario sentence generated by Data Associations [31]

While the <<gateway>> is mapped to a sentence in pre-condition use case description as represented in table2.

Gateway	Graphical representation	Originated Pre-condition.
Exclusive Decision		The <gateway condition> is <sequence flow condition>.
Parallel splitting		The <source name> has been completed.
Inclusive Splitting		The <sequence flow condition> is true.
Complex Splitting		The <sequence flow condition> is true.
Exclusive merging		The <source name> [exclusive or <source2 name>] has been completed.
Parallel join		The <source name> [and < source2 name>] has been completed.
Inclusive merging		The <source name> [or <source2 name>] has been completed.
Complex merging		The <source name> [or <source2 name>] has been completed.

Table 2:Pre-condition sentences generated by gateways [31]

The <<event>> is transformed to sentence in trigger use case description as shown in table3.

Catching Event	Originated sentence in use case trigger.
None	The event <event definition> occurs.
Message	The message <event definition> arrives from <source>.
Timer	The time-date <event definition> is reached.
Conditional	The condition <expression> become true.
Signal	The signal <event definition> arrives.
Multiple	The <event definition> [or <event definition>] occurs.
Parallel Multiple	The <event definition> [and <event definition>] occurs.

Table 3:Triggers generated by events [31]

The proposed approach in our thesis, BMSpec, builds on this approach [31]. BMSpec adopts some of their defined rules in transforming the business process model in to use case description, however it deals with some BPMN notations in different ways. The use case diagram in BMSpec is a rich diagram, it handles four types of associations between use cases (invoke, extend, include and precede) and handles post-condition and comment in use case description. The main difference, BMSpec achieves the transformation automatically, while [31] achieves transformation manually.

Another systematic approach [7] proposes “a method for deriving system requirements based on business process models”. This approach “integrates Requirements Engineering with Business Process Engineering and defines BORE: a Business-Oriented approach to Requirements Elicitation”. This approach is especially effective when system requirements are not fully known up front and must be discovered. They classify the difficulties of requirement engineering process in to “essential” and “accidental” difficulties. They then apply techniques and methods which help them avoid the accidental difficulties. They use three

different notations (BPMN, Map and UML) during requirements analysis and adopt EKD [9] approach to provide “a systematic and controlled way of analysing, understanding, developing and documenting an enterprise and its components” [7]. The BORE approach consists of three steps: First, As-Is Business Process Modelling. Second, Business Process Improvement. Third, Functional Requirements Elicitation. In BMSpec, we use the same concept of integrating Requirements Engineering with Business Process Engineering, however not in all cases, only when the business process model is (AS_Is) business model and the need to (business process model) reengineering is to define it for the software system purpose.

A similar approach is proposed by [3]. It describes business process modelling as a tool for successful definition of requirements specification, represented by use case diagram. They present the transformation process as a guideline, which consists of several patterns. These patterns consist of several steps, such as, steps to determine system actions, which will be provided by newly developed system, and human actions. Human actions are then excluded and then the business process model is rewritten.

In this section we represent two types of approaches that use task oriented to derive requirement specifications: automated approaches and manual approaches. The automated approaches, that transform business process model to use case diagram [1, 2, 6], have several limitations. They transform all types of processes (automated processes and manual processes), assuming all business processes are supposed to be computerized. However, this is not always true. Some processes are more suitable to be performed by hand and cannot be achieved by software systems. These approach lead to creating redundant and complex use case model. In addition, in cases when they specify all tasks as automated tasks, these approaches do not use business process reengineering, leading to inaccurate or incomplete resultant use cases. Another limitation, of these approaches, is neglecting the association between use cases, such as inclusion, extension or generalizations, so resultant use case diagrams need manual correction. However, these approaches use the concept of “mapping between two meta-models”, which

is the same concept we use in BMSpec, but we use a predefined set of heuristic rules to build the use case model. In addition, they use the activity model and RAD to represent business process models but we use BPMN.

The manual approaches [31, 7, 3] to transform business process model to use requirement specification address some limitations of the automated approaches, such as determining automated processes and associations between generated use cases. But still has two limitations: the transformation is done manually, which can take a lot of time and introduces high probabilities of arising errors. One of these manual approaches [31] attempts to generate use case descriptions, the others [7, 3] generate just use case diagram.

BMSpec takes benefit from the above approaches [31, 1, 2, 6], in employing two ways of transformation: automated transformation and the use of a set of predefined rules to construct well-defined business process model.

3.5 Discussion

In this literature review, we presented related work to deliver system requirement specification using business processes in an organization. These approaches could be classified into three main categories: Goal-oriented approaches, Organizational-oriented approaches and Task-oriented approaches. In each classification, we presented several relevant approaches and identified their strengths and limitations. We can conclude that Goal-oriented approaches offer a potential approach in determining the goal of the system to be developed, but it uses different notations which may result in sophisticated and crowded models, which are difficult to link with the next phases of software development process and difficult to understand by all stakeholders, thus are considered any further. On the other, despite that organizational-oriented approaches produce richer business models, which enable derivation of richer system requirements, and has the ability to represent most parts of an organizational model, it has a critical limitation, the effect of system goal on business processes is weakly addressed, and thus similarly are not considered any further for our work, i.e. for the proposed approach, BMSpec, in this thesis.

In this thesis, however, we adopt the task- oriented approaches to represent the business process model. Task-oriented approaches address the limitation of the Goal-oriented approaches (i.e. different notations and complex models) by using the BPMN, which is a standard notation language understandable by stakeholders. They also address the relevant limitation of the organizational-oriented approaches (i.e. weakly addresses the effect of system goal on business process). Table 4 summarises the analysis of the relevant considered task- oriented approaches. We are interested in two main features: the way of transformation and the richness of generated output. The “+” symbol, in Table 4, indicates that the respective approach supports or provides the noted capability, the “-” symbol indicates that it does not.

Approach	Automated Transformation	Output		
		Use case diagram	Association between use cases	Use case Description.
Dijkman et al, 2002 [1]	+	+	-	-
Odeh et al, 2003 [2]	+	+	-	-
Aburub et al, 2012 [6]	+	+	-	-
Cruz et al, 2014 [31]	-	+	-	+
Adam et al, 2014 [7]	-	+	+	-
Rhazali et al, 2014 [32]	-	+	+	-

Table 4: Analysis of task oriented approaches for deriving use case model.

In our approach we developed a set of rules to prepare a well-defined business process model. These rules are used to determine the purpose of the system by using business process re-engineering techniques. Then, we achieved transformation automatically by mapping between meta-model of business process objects and use case objects. This automatic transformation depends on a set of predefined heuristic rules. Some of these rules (14 rules) were extended

from those proposed in [31], however most of these rules (17 rules) were developed part of this work. To achieve automatic transformation, we developed a smart algorithm, which takes a business process model as input, in BPMN, and transform it, automatically, to generate a use case model using the proposed 36 heuristic rules. Our approach addresses the limitations of existing approaches (see Table 4), by determining four associations types between generated use cases in use case diagram, determining automated (system) and manual (human) processes and generating a richer use case model that consists of both the detailed use case description and the use case diagram.

Chapter 4

Proposed Approach: BMSpec

This chapter introduces the proposed systematic approach. It describes the main steps in the BMSpec: prepare a well-defined business process model, generate use case diagram and generate use case description.

4.1 Introduction.

This section proposes a new approach which derives requirement specification from business process model, we know the fact that “business process model is designed for business people while use case model is for system analysts or system developers” [29]. BMSpec defines that "Use Case Model" is inclusive of both the Use Case diagram and the detailed Use Case descriptions, and hypothesises that it could be generated from the business process model.

This approach consists of main three systematic steps as shown in figure7. First step, is preparing a well-defined business model. This step, depends on the status of the existing business process model, is a manual reengineering of the business model. Second step, is the automated transformation from business process model to UML use case model. In this step, BMSpec utilises an algorithm to map XML objects in between both business process and use case models, using a set of heuristic rules that define the transformation of objects and relations from business process model to use case diagram. Third step, is the automated transformation from business process model to use case description. In this step BMSpec uses another set of heuristic rules, and utilises another algorithm to map notations from the business process model to the detailed use case descriptions. Each of these steps, in BMSpec, will be discussed in details in the following subsections.

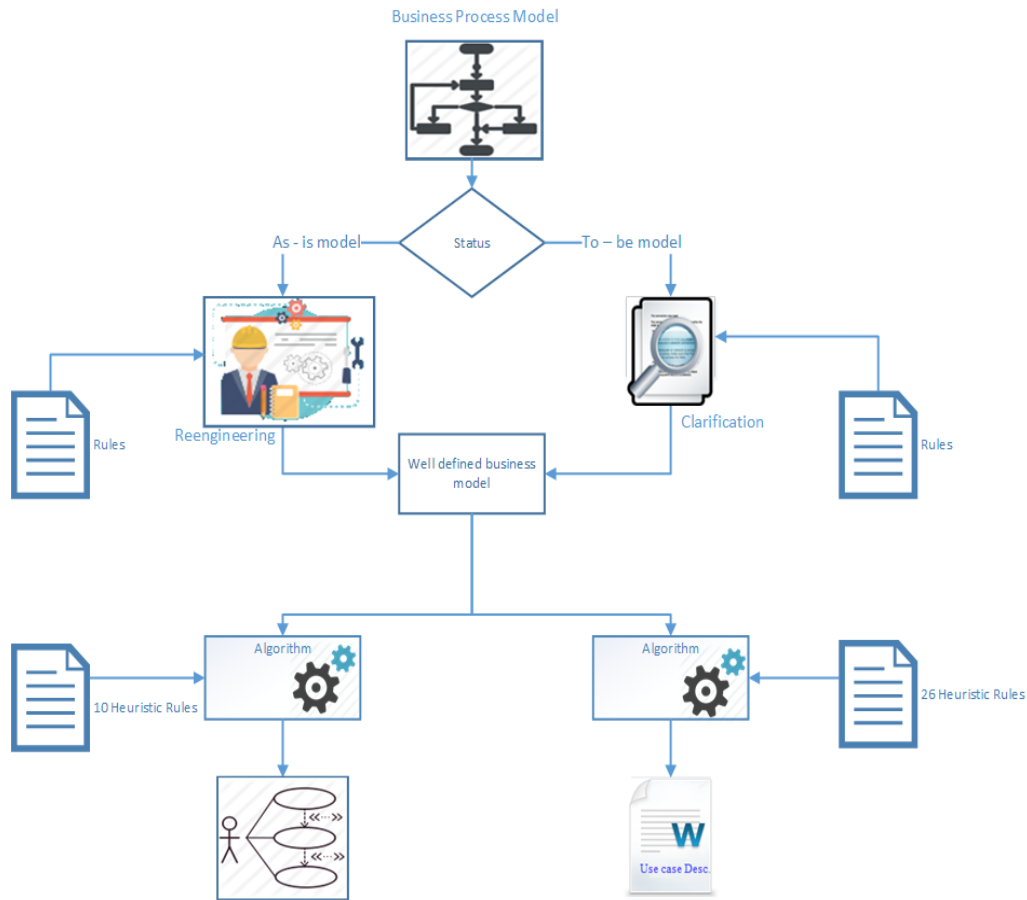


Figure 7: BMSpec.

4.2 First Step: Preparing a well-defined business process model.

This step represents a manual transformation of the existing business process model. This transformation can be a simple clarification or a business process reengineering depending on the status of the business process model, these two types of transformations are described below.

- **Business process Reengineering:** if the business model is just a manual As_ Is business process model, and does not cover the user interaction with the system, then the model needs reengineering in order to build (To_ Be) business process model. In this step we analyze the purpose of the system and determine the effect of the information system should have on the business processes (To-Be). So we are interested in the automated tasks that represent user's interactions with the system. "But automating processes for the aim of

automation does not lead to significant improvements” [13]. Thus, many research such as [14] suggest, instead of blindly automating manual processes, the processes are reengineered while taking advantages of the possibilities for automation. “Reengineering means reshaping the way business is done, to achieve this reshaping it is important to take an integrated look at both process and information flows simultaneously, focusing on how information is used in the process and how people interact with systems” [15].

In order to reengineer the business process model effectively, we should focus on both perspective of the system (asking what will make up a well-defined use case?) and business process perspective (asking what is needed from the IS?) [20] While well-defined use case must specify a functionality which an actor wants to achieve by using the system.

In BMSpec we should identify the tasks that represent interactions with the software system. Because only these tasks are important to generate use case model. Each well-defined use case must specify a functionality, which an actor wants to achieve by using the system. We developed the following set of rules to aid the transformation to improve an existing business process model to become a (To-Be) business process model:

- **Rule 1:** Define automated task that achieved fully by the system without any action from user as **service task**.
- **Rule 2:** Define the tasks represent an action of user on the system.
- **Rule 3:** Remove manual <<task>> that cannot be achieved by the system.
- **Rule 4:** Ensure <<gateway>> appropriately used.
- **Rule 5:** Specify <<events for task>> not only <<start events>> but also <<intermediate events >> and <<end events >>.
- **Rule 6:** Define all participants as Roles not as <<pool>> or <<lane>>.
- **Rule 7:** Assign appropriate (performer) for each user <<task>>.
- **Rule 8:** Specify required <<data objects>>.
- **Rule 9:** Specify required <<message flow>>.

- **Business model Simple clarification:** if the model already designed for the software system purpose, and the effect of the information system clear on the business processes (To-Be). But some of notation not clearly enough, such as specifying task type and declare condition for gateways and events name and type, then the model need to declare the notation depending on a set of rules that we created:
 - **Rule10:** Set task type as service task, If the task is fully executed by the system.
 - **Rule 11:** Set notation description.
 - **Rule11.1:** set a name for each <<gateway>>.
 - **Rule11.2:** Set a name for each <<condition>>.
 - **Rule11.3:** Set a name for each <<data object>>.
 - **Rule11.4:** Set a name for each <<data store>>.
 - **Rule11.5:** Set a name for each <<start event>>.
 - **Rule11.6:** Set a name for each <<intermediate event>>.
 - **Rule11.7:** Set a name for each <<end event>>.
 - **Rule11.8:** Set a name for each <<message flow>>.
- **Business model with none modification:** if the model already designed for the software system purpose, and the effect of the information system clear on the business processes (To-Be). Also, all notation is declared: such as determining service task, data object name, data store name, gateway name, event name, condition name and message name. In this case the business process model is a well-defined model and ready to go to the next step in order to generate requirement specification.

4.3 Second step: Generating Use Case Diagram.

In a well-defined business process model a task (activity) represents user interaction with the system. In use case diagram, “a use case presents a goal user wants to achieve by using the system” [26]. So we assume that each activity can be mapped to a use case.

We created a mapping between the two meta-models by evaluating their definitions.

“A Pool represents a graphical container for partitioning a set of activities from other Pools. The activities within separate Pools are considered self-contained Processes which are performed by same role” [11]. So we cannot map each pool to an actor as others approach do [31, 31].

In business process model “anyone who has an activity to perform, relevant to the process, is said to be a participant” [11]. In use case model, an actor represents a user of system. So we map each participant concept in business process model to Actor concept in UML use case model. Association between a task and a role to an association between the corresponding actor and use case. The output of BMSpec is a use case model in the following subsections we explain how to generate each item:

4.3.1 Generate Items in use case diagram

The use case diagram consist of use cases, actors and associations between them, in this section we explain how to generate these items using a set of heuristic rules, we proposed three heuristic rules to generate items in use case diagram

- **Rule1:** Map each activity to a use case. The use case name is the activity name as shown in Figure 8.



Figure 8: R1

- **Rule2:** Map each performer to an actor. The actor name is the role name as shown in Figure 9.



Figure 9:R2

- **Rule3:** Map each association between an activity and performer to association between actor and use case in use case diagram as shown in Figure10



Figure 10: R3

Notation Name	Graphic Notation	Generated Use case item
<<Activity >>		
Performer	<p>Type <input checked="" type="radio"/> Role <input type="radio"/> Entity</p> <p>A resource is a Business Entity (e.g., a company, company division, or a customer) or a Business Role (e.g., a buyer, a seller, a credit analyst), which controls or is responsible for a business process or a business activity.</p>	
Association between <<activity>> and performer	<p>Name <input checked="" type="radio"/> Activity1</p> <p>Description <input checked="" type="radio"/></p> <p>Performers <input checked="" type="radio"/> Participant</p>	

Table 5: Rules to generate use case diagram

4.3.2 Generate Associations in use case diagrams

- **Precede Association Between Two Use Cases:**

The precede association is used to show that base Use Case must complete before preceded Use Case can begin. It is a type of ordering the executing of use cases. We find the same behaviour in this case in business process model when an activity is connected to another activity by a sequence flow, in business process model the Sequence flow is used to display the order of executing activities in a process [24]. So we map sequence flow between two activities to a precede association between the use cases which represents that activities. As represented in Table 6.

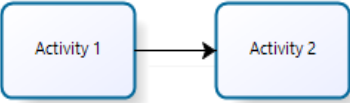
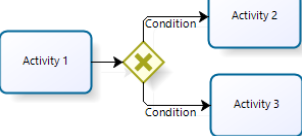





Notation Name	Graphic Notation	Use case Association
Sequence Flow between two activities		The use case which represents source activity precede the use case which represents target activity
Sequence flow between exclusive <<gateway>> and <<activity>>		The use case which represents target activity extends use case which represents source activity
Sequence flow between <<data object>> and <<activity>>		The use case which represents activity include receive data object name
Sequence flow between <<activity>> and <<data object>>		The use case which represents activity include send data object name
Sequence flow between <<data store>> and <<activity>>		The use case which represents activity include read information from data object name
Sequence flow between <<activity>> and <<data store>>		The use case which represents activity include write information on data object name
Sequence Flow between two activities, the target <<activity>> is <<service activity>>.		the corresponding use cases connect by a association "invoke"

Table 6: Rules to generate use case associations

Rule4: Map sequence flow between two activities to "Precede" association between the corresponding use cases as shown in Figure 11.



Figure 11: R4

- **The “<<Extend>>” Association Between Two Use Cases:**

The extend association is used to extend the base use case and adds more functionality to the system. But the extending use case is dependent on extended use case, it is usually optional and can be triggered conditionally. While the extended use case must be meaningful on its own and independent from extending use case [26].

We find the same behaviour of this case in business process model when an activity is connected to an exclusive gateway and the execution of the activity is optional and depend on a condition and make a decision from gateway. So we map sequence flow between activities and exclusive gateway to an extend association between the use cases which represents that activities. As represented in table 7.

Rule5: Map exclusive decision gateway between two activities to "extend" association between the corresponding use cases As shown Figure 12.



Figure 12: R5

- **The “<<Include>>” Association Between Two Use Cases:**

Include association used to show that the behaviour of included use case is part of including use case, but the base use case is not complete without the included use case, and the included use case is mandatory and not optional[26].

We find the same behaviour of this case in business process model when an activity is connected to data association, the activity includes another activity such

as read information from data store or write information on data store or send data object or receive data object. So we map data association between activity and data objects and data store to an include association between the base use cases which represents that source activity and read or receive or write or send information use case. As represented in table7.

Rule6: Map data association between activity and input data store to include association between the corresponding use cases as shown in Figure 13.



Figure 13: R6

Rule7: Map data association between activity and output data store to include association between the corresponding use cases as shown in an output data store as shown in Figure 14.



Figure 14: R7

Rule8: Map data association between activity and input data object to include association between the corresponding use cases as shown in an output data store as shown in Figure 15.



Figure 15: R8

Rule9: Map data association between activity and output data object to include association between the corresponding use cases as shown in an output data store as shown in Figure 16.



Figure 16: R9

Rule10: Map sequence flow between activity and service activity to Invoke association between the corresponding use cases as shown in Figure 17.



Figure 17: R10

4.4 Algorithm to generate use case diagram

Function GenerateDiagram()

- (1) Read business process model
- (2) **For** each A **in** Activities
 applyRule1(A)
End for
- (3) **For** each P **in** performer
 ApplyRule2(P)
End for
- (4) **For** each S **in** association
 ApplyRule3(S)
End For
- (5) **For** each S **in** sequence flow between activities
 ApplyRule4(S)
End For
- (6) **For** each E **in** exclusive decision gateway between two activities
 ApplyRule5(E)
End For
- (7) **For** each D **in** input data store
 ApplyRule6(D)
End For
- (8) **For** each D **in** output data store
 ApplyRule7(D)


```
        End For
    (9) For each D in input data object
        ApplyRule8(D)
    End For
    (10) For each D in output data object
        ApplyRule9(D)
    End For
    (11) For each S in sequence flow between activity and service activity
        ApplyRule10(S)
    End For
    (12) Write on use case diagram.
End Function
```

```
Function applyRule1(Activity A)
    A → UC
    A is activity
    UC is use case
```

End Function

```
Function applyRule2(Performer P)
    P → Ac
    P is Performer
    Ac is Actor
```

End Function

```
Function applyRule3(Association S)
    S → UCS
    S is Association between activity and performer in business model
    UCS is Association between use case and actor in use case diagram
```

End Function

```
Function applyRule4(sequence flow S)
    Create Precede Association PA
    S is sequence flow between activities
    PA is Precede Association in use case diagram
```

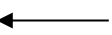
End Function

```
Function applyRule5(exclusive decision gateway E)
    Create Extend Association EA
    E is exclusive decision gateway between two activities
```

EA is Extend Association in use case diagram

End Function

Function applyRule6(input data store D)

UC  "Reads information from" + D.name

Create Include Association IA

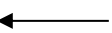
D is input data store

UC is use case

IA is Include Association in use case diagram

End Function

Function applyRule7(output data store D)

UC  "Writes information on" + D.name

Create Include Association IA

D is output data store

UC is use case

IA is Include Association in use case diagram

End Function

Function applyRule8(input data object D)

UC  "Receive " + D.name

Create Include Association IA


D is input data object

UC is use case

IA is Include Association in use case diagram

End Function

Function applyRule9(output data object D)

UC  "Send " + D.name

Create Include Association IA

D is output data object

UC is use case

IA is Include Association in use case diagram

End Function

Function applyRule10(sequence flow S)

Create Invoke Association IA

IA is Invoke Association in use case diagram

S is sequence flow between activity and service activity

End Function

4.5 Third Step: Generating Use Cases Description

This section explains how to generate use case description from business process model represented in BPMN using a set of heuristic rules some of these rules have been proposed in [31] we extend them and add new ones.

Various templates have been suggested for the textual description of use cases [73, 74, 75], each template has many items to be described, but there are a common items between these three templates such as: use case name, actor, trigger, precondition, description, and scenario. In BMSpec we use the same simplified template used in [31] this template based on a simplification of the template [73]. It is composed of 7 fields, which are named and described in [figure 18](#).

Use Case name	“The use case name identifies the goal as a short active verb phrase”.
Actors	“List of actors involved in the use case”
Pre-Conditions	“Conditions that must hold or represent things that happened before the use case starts.”
Post-Conditions	“Conditions that must hold at the conclusion of the use case.”
Trigger	“Event that starts the use case.”
Scenario	“Sequence of interactions describing what the system must do to move the process forward.”
Comment	Comment on use case.

Figure 18: adopted Use case description template [31]

4.5.1 Data Association:

It is very often when executing a business process, to exchange data either during the process or after the end of process. Data associations are usually used to transfer data between activities and data objects or data store [24]. A successful execution of a process may produce data, but this data can be modeled by several types of data objects such as: data object, text annotations and data stores.

Table 7 represents the output sentences that will be generated in the scenario field depending on data association type.





Notation Name	Graphic Notation	Scenario
Input <<data object>> represents the source in this association, while the <<activity>> represents target		“Receives Data Object Name”
Output <<data object>> represents the target in this association, while the <<activity>> represents source		“Sends Data Object Name”
Input <<data store>> represents the source in this association, while the <<activity>> represents target		“Reads information from Data Store Name”
Output <<data store>> represents target in this association, while the <<activity>> represents source		“Writes information on Data Store Name”

Table 7: Rules to transform data association

Rule11: Map data association between input data store and activity to sentence “Reads information from Data Store Name”, added to the scenario.

Rule12: Map data association between output data store and activity to sentence “Write information on Data Store Name”, added to the scenario.

Rule13: Map data association between input data object and activity to sentence “Receive Data Object Name”, added to the scenario.

Rule14: Map data association between output data object and activity to sentence “Send Data Object Name”, added to the scenario.

4.5.2 Artifacts:

“Artifacts used to show additional information about a Process, this information can not directly related to the Sequence Flows or Message Flows of the Process” [24]. One of artifact type is text annotation, it enables modeler to provide additional information for the reader of a BPMN Diagram, and at the same time it does not affect the flow of the process.

The sentence generated by text annotation is presented in Table 8, It will be added to the comment field.


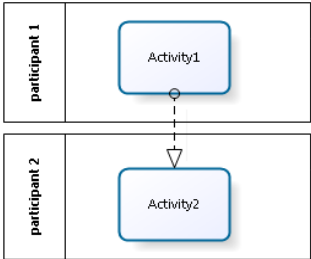
Notation Name	Graphic Notation	Sentence in Use Case comment
Annotation Association used to link text <<annotations>> to an <<activity>>.		Same Text in the Annotation

Table 8: The sentence generated by text annotation.

Rule15: Map data association between text annotation and activity to sentence contains the same text in the annotation, added to the comment.

4.5.3 Message Flow:

In Business process model notation, message flow used to exchange messages between different pools. Message flow also used to show the flow of messages between pools, it is graphically represented in dotted line with an arrow head. The sentence generated by message flow is represented in Table 9.

Notation Name	Graphic Notation	Sentence in Scenario
Output <<message flow>>		“Source activity Sends message name to target participant name.”

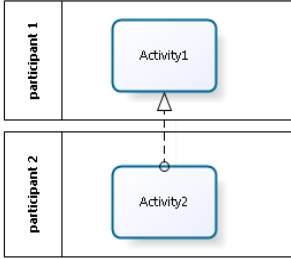
Input <<message flow>>		<p>“Source activity Receives message name from participant name.”</p>
------------------------	---	---

Table 9: sentence generated by message flow.

It depends on the direction of the message, if the activity receives the message (input message) or the activity send a message (output message). The sentences added to scenario.

Rule16: Map input message flow connected to an activity to sentence “receive message name from sender name”, added to the scenario.

Rule17: Map output message flow connected to an activity to sentence “send message name to receiver name”, added to the scenario.

4.5.4 Sequence Flow between two activities:

Sequence flow is used to display the order of executing activities in a process. It is used also to connect flow elements (activities, events and gateways) within the same pool: either within the same pool/lane, or across lanes in the same pool [24]. Sequence Flow is graphically represented as a solid line with an arrowhead. The sentence generated by sequence flow between two activities represented in Table 10. Added to the Pre-condition. While the sequence flow with gateways and events will be displayed in the following subsections.

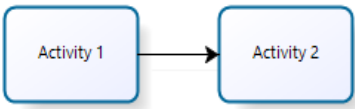
Notation Name	Graphic Notation	Sentence in Pre-condition.
Sequence Flow between two activities		<p>“The source activity name has been completed.”</p>

Table 10: sentence generated by sequence flow between two activities.

Rule18: Map sequence flow between two activities to sentence “The source activity name has been completed”, added to the Pre-Condition.

4.5.5 Sequence Flow between gateway and activities

Gateways are responsible for controlling how a business process flows. The work to do in a process and the output of that process may vary under different conditions. These conditions are evaluated using gateways and the decision made upon these conditions [24]. The gateway may use as diverging (splitting gateway) or converging (merging gateway), the difference between these gateways is the “splitting gateways have one incoming sequence flow and two or more outputs sequence flow” [24]. While “merging gateways have two or more incoming sequence flow and one output sequence flow” [24]. The sentence generated by sequence flow between gateways and activities are shown in Table 11.

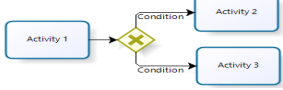

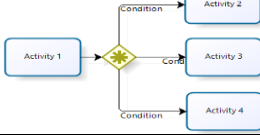
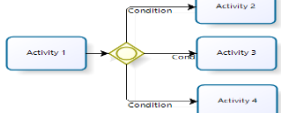
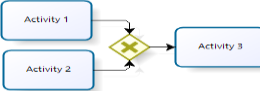


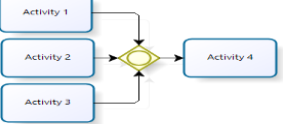
Notation Name	Graphic Notation	Sentence in Pre-condition.
<<Exclusive Decision Gateway>>		“The gateway condition is sequence flow condition.”
<<Parallel splitting Gateway>>		“The source Activity name has been completed.”
<<Complex Splitting Gateway>>		“The sequence flow condition is true.”
<<Inclusive Splitting Gateway>>		“The sequence flow condition is true.”
<<Exclusive merging Gateway>>		“The activity1 exclusive or activity2 name has been completed.”
<<Parallel join Gateway>>		“The activity1 and activity2 has been completed.”
<<Complex merging Gateway>>		“The activity1 or activity2 or source3 name has been completed.”
<<Inclusive merging Gateway>>		“The source name or source2 name or source3 name has been completed.”

Table 11: The sentence generated by sequence flow between gateways and activities

The sentences will be appended to the Pre-Condition of the use case description of the use case that represents the target activity. While the “gateways output sequence flow may have condition that allows to select a specific path” [24]. Each output sequence flow generates a pre-condition.

Rule19: Map sequence flow between exclusive decision gateway and activity to sentence “The gateway condition is sequence flow condition”, added to the Pre-Condition.

Rule20: Map sequence flow between parallel splitting gateway and activities to sentence “The source Activity name has been completed”, added to the Pre-Condition.

Rule21: Map sequence flow between complex splitting gateway and activity to sentence “The sequence flow condition is true”, added to the Pre-Condition.

Rule22: Map sequence flow between inclusive splitting gateway and activity to sentence “The sequence flow condition is true”, added to the Pre-Condition.

Rule23: Map sequence flow between exclusive merging gateway and activities, to sentence “The source activity 1 name exclusive or source activity 2 name has been completed”, added to the Pre-Condition.

Rule24: Map sequence flow between parallel join gateway and activities to sentence “The source activity 1 and source activity 2 has been completed”, added to the Pre-Condition.

Rule25: Map sequence flow between complex merging gateway and activities, to sentence “The source activity 1 or source activity 2 ... or source activity 3 has been completed”, added to the Pre-Condition.

Rule26: Map sequence flow between inclusive merging gateway and activities to sentence “The source activity 1 or source activity 2 ... or source activity 3 has been completed”, added to the Pre-Condition.

4.5.6 Sequence Flow between <<Event>> and <<Activity>>

Events are something that happens and may have impacts on a business process [24]. They can influence the executing path and time of activities. Three main types of events: End Event which indicates where the process ends [24], Start

Event which indicates when a process will start, Intermediate event which indicates where something happens somewhere between the start and end.

Any process should have start event to indicate when the process begins, and end event to indicate when the process complete. While intermediate event determine the business flow, it can be attached to an activity for modelling an event which may happen during the execution of the activity and also it can be connected by sequence flow to show an event that happens after the execution of the activity. Events are graphically represented as circles. Sometimes, there are icons within the circles to represent the type of the event trigger [24].

The sentence generated by sequence flow between events and activities are shown in Table 12, added to the trigger.




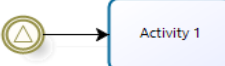

Notation Name	Graphic Notation	Trigger
<<Start Event>>		The event name occurred
<<Intermediate Event (None) >>		The event name occurs.
<<Intermediate Event (Timer) >>		“The time-date event name is reached.”
<<Intermediate Event (Signal) >>		The signal event name arrives.
<<Intermediate Event (Message) >>		The message event name arrives from source.

Table 12: The sentence generated by sequence flow between events and activities

Rule27: Map sequence flow between start event and activity to sentence “The event name occurred”, added to the Trigger.

Rule28: Map sequence flow between Intermediate Event (None) and activity to sentence “The event name occurs”, added to the Trigger.

Rule29: Map sequence flow between Intermediate Event (Timer) and activity to sentence “The time-date event name is reached”, added to the Trigger.

Rule30: Map sequence flow between Intermediate Event (Signal) and activity to sentence “The signal event name arrives”, added to the Trigger.

Rule31: Map sequence flow between Intermediate Event (Message) and activity to sentence “The message event name arrives from source activity”, added to the Trigger.

4.5.7 Sequence Flow between Activities and Events

The sequence flow between activity and end event is transformed as post-condition because when the activity executed the end event will execute directly. Also the sequence flow between activity and intermediate event is transformed to a post-condition.

The sentence generated by sequence flow between activity and event shown in Table 13. Added to the Post-Condition.


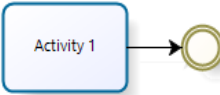

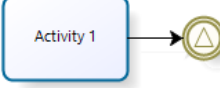
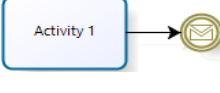
Notation Name	Graphic Notation	post-condition
<<End Event>>		“The event name is created. The process ends”
<<Intermediate Event (None)>>		The event name event occurs
<<Intermediate Event (Timer)>>		The time-date event name event is created.
<<Intermediate Event (Signal)>>		“The Signal event name is send”
<<Intermediate Event (Message)>>		“The Message event name is send”

Table 13: The sentence generated by sequence flow between activities and events.

Rule32: Map sequence flow between activity and End event to sentence “The event name is created. The process ends”, added to the post-condition.

Rule33: Map sequence flow between activity and Intermediate Event (Timer) to sentence “The time-date event name event is created”, added to the post-condition.

Rule34: Map sequence flow between activity and Intermediate Event (None) to sentence “The event name event occurs”, added to the post-condition.

Rule35: Map sequence flow between activity and Intermediate Event (Message) to sentence “The Message event name is send”, added to the post-condition.

Rule36: Map sequence flow between activity and Intermediate Event (Signal) to sentence “The Signal event name is send”, added to the post-condition.

4.6 Algorithm to generate use case description

Function GenerateDescription()

(1) Read business process model

(2) **For each** D **in** input data store

S ← “Reads information from” + D.name

D is input data store

S is scenario

End For

(3) **For each** D **in** output data store

S ← “writes information on” + D.name

D is output data store

S is scenario

End For

(4) **For each** D **in** input data object

S ← “Recieve” + D.name

D is output data store

S is scenario

End For

(5) **For each** D **in** output data object

S ← “Send” + D.name

D is output data store

S is scenario

End For

(6) **For** each A **in** annotation

C ← A.name

A is Annotation

C is comment

End For

(7) **For** each M **in** InMessage Flow

S ← “receive message” + M.name

M is In message flow

S is scenario

End For

(8) **For** each M **in** OutMessage Flow

S ← “send message” + M.name

M is out message flow

S is scenario

End For

(9) **For** each S **in** sequence flow between activities

P ← “send message” + M.name

S is sequence flow between activities

P is pre-condition

End For

(10) **For** each G **in** exclusive decision gateway between two activities

P ← “The” + source activity + “has been completed”

G is exclusive decision gateway between two activities

P is pre-condition

End For

(11) **For** each G **in** parallel Join gateway between two activities

P ← “The” + source activity + “has been completed”

G is parallel Join gateway between two activities

P is pre-condition

End For

(12) **For** each G **in** complex merging gateway between two activities

P ← "The" + source activity + "has been completed"
G is complex merging gateway between two activities
P is pre-condition
End For

(13) **For** each G **in** inclusive merging gateway between two activities
P ← "The" + source activity + "has been completed"
G is inclusive merging gateway between two activities
P is pre-condition
End For

(14) **For** each S **in** sequence flow between start event and activity
T ← " The " + event name + "occurred"
S is sequence flow between start event and activity
T is Trigger
End For

(15) **For** each S **in** sequence flow between Intermediate Event (None) and activity
T ← " The " + event name + "occurred"
S is sequence flow between Intermediate (None) event and activity
T is Trigger
End For

(16) **For** each S **in** sequence flow between Intermediate Event (Timer) and activity
T ← = " The timedate" + event name + "is reached"
S is sequence flow between Intermediate (timer) event and activity
T is Trigger
End For

(17) **For** each S **in** sequence flow between Intermediate Event (Message) and activity
T ← = " The " + event name + "arrives"
S is sequence flow between Intermediate (message) event and activity
T is Trigger
End For

(18) **For** each S **in** sequence flow between activity and end event
P ← = " The " + event name + "created. The process ends"

S is sequence flow between activity and end event

P is Post-condition

End For

(19) **For** each S **in** sequence flow between activity and Intermediate (None) event

P ← = “ The” + event name + “occurs”

S is sequence flow between activity and Intermediate Event (None) event

P is Post-condition

End For

(20) **For** each S **in** sequence flow between activity and Intermediate (Timer) event

P ← = “ The time-date ” + event name + “is created”

S is sequence flow between activity and Intermediate Event (Timer) event

P is Post-condition

End For

(21) **For** each S **in** sequence flow between activity and Intermediate Event (Signal) event

P ← = “ The” + event name + “is send”

S is sequence flow between activity and Intermediate Event (signal) event

P is Post-condition

End For

(22) **For** each S **in** sequence flow between activity and Intermediate Event (Message) event

P ← = “ The” + event name + “is send”

S is sequence flow between activity and Intermediate Event (message) event

P is Post-condition

End For

(23) Write on use case Description.

End function

Chapter 5

Evaluation

This chapter describes the evaluation methodology of BMSpec, business process model selection, evaluation method, experiment design.

5.1 Evaluation Methodology.

This section describes the evaluation methodology used to evaluate BMSpec. Based on the work of [83] on model evaluation, we are using these relevant criteria: Sufficiency, Efficiency and correctness:

- 1- **Sufficiency evaluation:** used to evaluate how much BMSpec cover sufficient number of BPMN notation types to get sufficiently representative or meaningful use case model.
- 2- **Efficiency evaluation:** used to evaluate the efficiency of BMSpec in transforming an input business model in utilising its notation richness. It does so by evaluating the effect of the level of richness of the input business process model, in terms of all its notations, and observes how well BMSpec makes use of or utilises them to generate a meaningful use case model.
- 3- **Correctness evaluation:** used to validate the correctness of BMSpec and its accuracy in generating a correct use case model. It does so by comparing resultant generated use case models against same use case models generated (by experts) using traditional manual requirement elicitation techniques.

The evaluation was conducted in two stages: conduct dry run experiments based on the evaluation methodology by running BMSpec on three arbitrary business process models to check if the experiment design works well; conduct wet run experiments based on the evaluation methodology by running BMSpec on 37 business process models to evaluate the correctness, sufficiency and efficiency of the approach.

The following sections describe the details of the evaluation methodology.

5.1.1 Business process model selection

In order to evaluate the performance of BMSpec, a group of experiments will be run on 37 business process models, where these models will be used to evaluate efficiency, sufficiency and correctness of developed approach.

BMSpec aims to reach or to generate requirement specifications more quickly than using traditional requirement engineering techniques, and get more accurate and more related to business needs requirement. 37 business process models are selected, seven models are used to show the correctness of developed approach and compared the results with traditional requirement engineering techniques. 30 models are used to evaluate the sufficiency and efficiency of BMSpec, 7 models are used to evaluate its correctness. These three types of evaluation and the models used in each evaluation, are described in details in the following sections:

5.1.2 Efficiency evaluation:

In order to evaluate the efficiency of BMSpec, we need to use different business process models with different level of efficiency, to achieve this we use 30 business process models, classify into six levels of efficiency as follows: 4 types of BPMN notation can affect the level of efficiency of the generated use case diagram: data association, decision gateway, service task and sequence flow between two activities. While five types of BPMN notation can affect the level of efficiency of generated use case description: data association, gateways, events, message flow and sequence flow between two activities.

So we proposed level of richness of the input business process model:

❖ **Level 1 Of Richness (LOR1):** these models contain rich types of business process, it contains data association, decision gateway, events, message flow, service task and sequence flow between two activities. 2 business process models classified as LOR1: OF Richness as shown in Table 14: LOR1 OF Richness.

Business process model	Decision Gateway.	Data Association.	Service task	Sequence flow between activities	Event.	Message flow
Open Vacancy	2	7	1	3	4	1
Invoice receipt	2	1	1	2	3	0

Table 14: LOR1 OF Richness

❖ **Level 2 Of Richness (LOR2):** contains, decision gateway, events, message flow, data association and sequence flow between two activities. While service task disappeared here no invoke relation, regarding that service task affects only invoke relation, this level is low richness than LOR1. 8 business process models classified as LOR2: OF Richness as shown in Table 15.

Business process model	Decision Gateway.	Data Association.	Service task	Sequence flow between activities	Event.	Message flow
Purchase via email	2	5	0	3	4	4
Order to cash	1	4	0	3	6	2
Auction	1	2	0	1	3	10
Water purchase	1	4	0	2	3	2
Travel agency	1	4	0	1	2	6
Request to proposal	4	2	0	2	6	0
Collect parts order	3	7	0	1	3	3
Claim assignment	2	10	0	2	3	0

Table 15: LOR2 OF Richness

Level 3 Of Richness (LOR3): contains service task, sequence flow between two activities, events, decision gateway, and message flow. While data association disappeared here which means this level is low richness than LOR1 and a little less than L2 because data association affect both scenario in use case description

and include relation in use case diagram. 5 business process models classified as LOR3: OF Richness as shown in Table 16.

Business process model	Decision Gateway.	Data Association.	Service task	Sequence flow between activities	Event.	Message flow
Cab booking process	2	0	2	1	5	6
book-selling-process	1	0	2	1	3	6
Customer pricing scenario	3	0	0	3	9	4
Leather manufacturing	2	0	0	2	2	4
Authoring process	1	0	0	1	3	4

Table 16: LOR3 OF Richness

❖ **Level 4 Of Richness (LOR4):** contains sequence flow between two activities, events, and message flow and data association. While decision gateway disappeared here which means this level is low richness than LOR1 and a little less than L2 because decision gateway affect also on precondition in use case description, also similar to LOR3. 4 business process models classified as LOR4: OF Richness as shown in Table 17.

Business process model	Decision Gateway.	Data Association.	Service task	Sequence flow between activities	Event.	Message flow
Manufacturing	0	2	0	3	5	2
license lifecycle App.	0	5	1	6	9	0
Insurance company offer	0	4	0	2	2	2
Collect vote	0	4	0	1	8	0

Table 17: LOR4 OF Richness.

Level 5 Of Richness (LOR5): contains sequence flow between two activities events, decision gateway. While data association and message flow disappeared here which means this level is low richness than LOR1, L2, LOR3, and LOR4 because data association affect the include relation in use case diagram and scenario in use case description, also message flow affects scenario, in this case there will be no scenario and no include relation. 5 business process models classified as LOR5: OF Richness as shown in Table 18.

Business process model	Decision Gateway.	Data Association.	Service task	Sequence flow between activities	Event.	Message flow
Credit card issuance	1	0	0	1	5	0
Hardware retailer	1	0	0	1	2	0
Loan request	2	0	0	1	2	0
Employee expenses approval	3	0	1	2	3	0
Class registration	3	0	0	1	5	0

Table 18: LOR5 OF Richness

❖ **Level 6 Of Richness (LOR6):** some models contains events, decision gateway and data association without a service task, sequence flow between two activities, and message flow, which means this level is low richness than LOR1, LOR2, LOR3, LOR4 and LOR5 because no invoke relation, no precede relation in use case model, also the precondition and scenario will be affected by eliminating sequence between activities and message flow. Some models have sequence flow between activities, but without data association or decision gateway, which means no extend relation, no include relation and eliminating pre-condition and scenario. 6 business process models classified as LOR6: OF Richness as shown in Table 19.

Business process model	Decision Gateway.	Data Association.	Service task	Sequence flow between activities	Event.	Message flow
Account creation	1	1	0	0	3	0
Working group	1	1	0	0	3	0
Leave request	1	0	0	0	2	0
Travel Plan Quote Process	0	0	0	1	5	0
Employee assignment	0	0	0	2	2	0
Online shop	0	0	1	2	2	3

Table 19: LOR6 OF Richness.

Depending on the results we will determine: which level of efficiency makes BMSpec works best, which level of efficiency makes it moderately works, but not best, and which level of efficiency makes it not working.

5.1.3 Sufficiency Evaluation

In order to evaluate how much BMSpec cover sufficient notation to get the appropriate use case model. We need to use a different business process model with diverse types of notation, and check for each relation between BPMN in each model if it is covered in BMSpec or not, and if not how it can affect the generated use case model. So we use the 30 business models as input to developed approach and observe all types of relations between BPMN. In BPMN we have main notations: data associations, message flow, sequence flow, associations, gateways, Events and activities.

The covered relations between notations is:

- ❖ Relation between <<activity>> and data associations.
 - 4 types of relation
- ❖ Relation between <<activity>> and message flow.
 - 2 types of relations

- ❖ Relation between <<activity>> and associations.
 - 1 type of relation
- ❖ Relation between <<activity>> and gateways.
 - 8 types of relation
- ❖ Relation between <<activity>> and Events.
 - 10 types of relation
- ❖ Relation between activities.
 - 2 types of relation

These 27 covered relation between notations represented in the 36 heuristic rules that we explained in chapter 4.

We are interested with notation that is directly connected to the activity because it will affect the output use case model, but some relation may exist between other notations such as relation between gateways, events, these relations need investigation to know if they affect the output use case model.

5.1.4 Correctness Evaluation

In order to evaluate BMSpec and prove its correctness and accuracy, we use a gold standard test, while the gold standard test is used to refer to the most accurate test, in our case the gold standard is using a traditional requirement elicitation technique to build a use case model and compare it with the generated use case model from BMSpec. We use seven different cases as gold standard, a detailed description of these cases exists in the iii. Appendix.

5.1.5 Evaluation Method

We conduct a dry run for the experiment to ensure that the experiment setup is working as designed before start the wet run. In the dry run we choose randomly three different business process model, use them as input to the automated algorithm and notice the generated use case model, we get a correct initial result. As we mention before we will conduct three types of evaluation: correctness evaluation, sufficiency evaluation and efficiency evaluation.

In order to know exactly how to measure the efficiency of BMSpec and how the richness of the input business process model affect the efficiency, we build a very rich business model, this model contains all notations that we covered in BMSpec: data association, message flow, service task, sequence flow between activities, sequence flow between gateway and activities, sequence flow between intermediate event and activities. Use this model to generate use case model, the results will notice and discuss how much its rich.

Then we change the richness of the business process model, we reduce its richness by eliminating some notation and notice the richness of the output use case model. The forty business process model will be used as input to BMSpec. Depending on the status of each model it will go in a specific type of manual transformation, two types of manual transformation are proposed: First, Business process reengineering to build (To_Be business process model). Second, simple business process clarification in order to clarify some notations and character some activities. After preparing the models for the next automated transformation, the models will input to the algorithm which convert the business process models to use model. For each model we will notice all inputs BPMN, and the output generated use case model all the elements such as: number of generated use case, actors, extend associations, include associations, precede associations, invoke associations, use case description, use case pre-condition, use case trigger, use case scenario, and use case comment.

5.1.6 Experiment Design

The automated algorithm implemented in VB.NET 2012 programming language running on a personal computer with windows 7 operating system and (i5) with 2.5 GHZ processor and 8GB memory.

We use Bizagi modeller version 3.0.0.022 for modelling the business process models.

There are many existing business process modelling tools, but we chose Bizagi for many reasons [19]: first, Bizagi handles the complete life cycle of a business process: Model, Automate, Execute, and Improve. Second, enable draw business

process of the system and export and import the process in many formats such as XPD. Third, it uses BPMN which it's standard language. Fourth, offers a simple graphical environment which enables business analysts to define and manage business rules. While the tools used for modelling use case diagram is enterprise architect.

After we represent each business process model using Bizagi we export the model as XML file, the XML file contains all the objects with represent the business model.

The automated algorithm reads the XML file which generated from Bizagi and transform it to a use case diagram XML file and Microsoft word file contains use case description file.

The generated XML use case model is structured as the XML schema supported by enterprise architect schema and the data type definition (DTD) file. We import the generated XML use case file into enterprise architect and view the diagram. The result use case diagram and Microsoft word use case description file are noticed and recorded as results to be analysed later.

Chapter 6

Results, Analysis and Discussion

This chapter aims to show the results of BMSpec with respect to three key characteristics, efficiency, sufficiency and correctness, these results will be analysed and discussed in details in the following sections:

6.1 Efficiency evaluation results

We start the evaluation by building a high rich business process model, high rich model means here it contains all the covered notations in BMSpec. We build it because we may not able to find a real business process model contains all the BPMN that we covered.

We choose an existing business process model and add the missing notation. The selected business process model is the open vacancy model, as shown in [Table 20: notation in open vacancy model](#)

, the existing notations and added notations, we use here only one type of gateway: decision gateway. Also the intermediate event, we choose three types instead of adding all types because it gives us a similar transformation concept.

Notation Name	existing	added
Activity	yes	
Performer	yes	
Service activity	yes	
Sequence Flow between two activities	yes	
Exclusive Decision Gateway	yes	
Input Data object	yes	
Output Data object	yes	
Input Data store	yes	
Output Data store	yes	
Annotation		yes
Input message flow		yes
Output message flow	yes	
Start Event	yes	
Intermediate Event (None)	yes	
Intermediate Event (Timer)		yes
Intermediate Event (Message)		yes
End event	yes	

Table 20: notation in open vacancy model

After building this high rich model we input it to BMSpec and notice efficiency of the output use case model, then in each iteration of running experiment we eliminate some notation and also notice the efficiency of the output. The result of this evaluation is displayed in Table21.

Notation richness	Input Business process model Notation																Output use case diagram element				Output use case description element								
	Activitiy Num.	Performer Num.	Service activity Num.	Sequence Flow between two activities Num.	Exclusive Decision Gateway Num.	Input Data object Num.	Output Data object Num.	Input Data store Num.	Output Data store Num.	Annotation Num.	Input message flow Num.	Output message flow Num.	Start Event Num.	Intermediate Event (None) Num.	Intermediate Event (Timer) Num.	Intermediate Event (Message) Num.	End event Num.	Use Case Num.	Actors Num.	Extend relation Num.	Include relation Num.	Precede relation Num.	Invoke relation Num.	Use Case Desc. Num.	Use Case Pre-Condition Num.	Use Case Post-Condition Num.	Use Case Trigger Num.	Use Case Scenario sentence Num.	Use Case Comment Num.
Full Notation	7	3	1	1	2	2	3	1	1	1	1	1	2	1	1	1	2	15	3	4	7	1	1	8	5	4	4	10	1
Without service activity	8	3	0	1	2	2	3	1	1	1	1	2	1	1	1	1	2	15	3	4	7	1	0	8	5	4	4	10	1
Without sequence flow between two activity	8	3	0	0	2	2	3	1	1	1	1	2	1	1	1	1	2	15	3	4	7	0	0	8	4	4	4	10	1
Without decision gateway	8	3	0	0	0	2	3	1	1	1	1	2	1	1	1	1	2	15	3	0	7	0	0	8	0	4	4	10	1
Without Data object	8	3	0	0	0	0	0	1	1	1	1	2	1	1	1	1	2	10	3	0	2	0	0	8	0	4	4	5	1
Without Data store	8	3	0	0	0	0	0	0	0	1	1	2	1	1	1	1	2	8	3	0	0	0	0	8	0	4	4	3	1
Without Annotation	8	3	0	0	0	0	0	0	0	0	1	2	1	1	1	1	2	8	3	0	0	0	0	8	0	4	4	3	0
Without message flow	8	3	0	0	0	0	0	0	0	0	0	0	0	1	1	1	2	8	3	0	0	0	0	8	0	4	4	0	0
Without start event	8	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2	8	3	0	0	0	0	8	0	4	3	0	0
Without Intermediate event	8	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	8	3	0	0	0	0	8	0	2	0	0	0
Without end event	8	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	3	0	0	0	0	8	0	0	0	0	0

Table 21: The effect of model richness on the efficiency of BMSpec.

We can notice from Table 21, in each iteration if we reduce the richness of the model, the efficiency of BMSpec will be reduced, so the relation between the richness of the input model and the efficiency of BMSpec is linearly increasing.

We can explain this relation as follows:

Eliminating the service activity cause losing the invoke association between use cases

As shown in Table 21, when we eliminate one service activity and convert it to normal activity, the invoke association in the use case diagram disappeared. But the use case description does not affected, and this is logic and related to **Rule10**.

Eliminating sequence flow between two activities cause losing the precede association between use cases and some pre-condition in use case description.

As shown in Table 21, when we eliminate one sequence flow between two activities, the precede association in the use case diagram disappeared. Also one pre-condition from use case description is loosed, and this is logic and related to **Rule4** and **Rule18**.

Eliminating sequence flow between decision gateway and activities cause losing the extend association between use cases and some pre-condition in use case description.

As shown in Table 21, when we eliminate two decision gateway, which connected with four activities, each one is connected with two activities, four extends association in the use case diagram disappeared. Also four pre-condition from use case description is loosed, and this is logic and related to **Rule5** and **Rule19**.

Eliminating data objects cause losing the include association between use cases and some scenario sentences in use case description.

As shown in Table 21, when we eliminate five data objects, five include association in the use case diagram disappeared. Also five scenario sentence from use case description is loosed, another losing occurs in number of use case, five system use case lost. And this is logic and related to **Rule8**, **Rule9**, **Rule13** and **Rule14**.

Eliminating data store cause losing the include association between use cases and some scenario sentences in use case description.

As shown in Table 21, when we eliminate two data store, two include association in the use case diagram disappeared. Also, two scenario sentence from use case description is loosed, another losing occurs in number of use case, two system use case lost. And this is logic and related to **Rule6**, **Rule7**, **Rule11** and **Rule12**.

Eliminating the Annotation cause losing the comment in use case description

As shown in Table 21, when we eliminate one annotation, the comment in the use case description disappeared. But the use case diagram does not affected, and this is logic and related to **Rule15**.

Eliminating the message flow cause losing some scenario sentences in use case description

As shown in Table 21, when we eliminate three message flows, three scenario sentences in the use case description disappeared. But the use case diagram does not affected, and this is logic and related to **Rule16** and **Rule17**.

Eliminating the start event cause losing some triggers in use case description

As shown in Table 21, when we eliminate one start event, one trigger in the use case description disappeared. But the use case diagram does not affected, and this is logic and related to **Rule27**.

Eliminating the intermediate event cause losing some triggers and post-condition in use case description

As shown in Table 21, when we eliminate three intermediate events, three triggers in the use case description disappeared and two post-condition, just two because one of these event not connected between two activities. But the use case diagram does not affected, and this is logic and related to **Rule28**, **Rule29**, **Rule30**, **Rule33**, **Rule34**, **Rule25** and **Rule36**.

Eliminating the end event cause losing some post-condition in use case description

As shown in Table 21, when we eliminate two end event, tow post-condition in the use case description disappeared, but the use case diagram does not affected, and this is logic and related to **Rule32**.

As we notice the richness of the business process model affects directly the efficiency of BMSpec. When we eliminate most of covered notation we get a poor use case model without any type of associations between use cases. Also, we get a poor use case description without triggers, preconditions, post-conditions, scenarios, and comments as shown in Figure 19 and Figure 20.

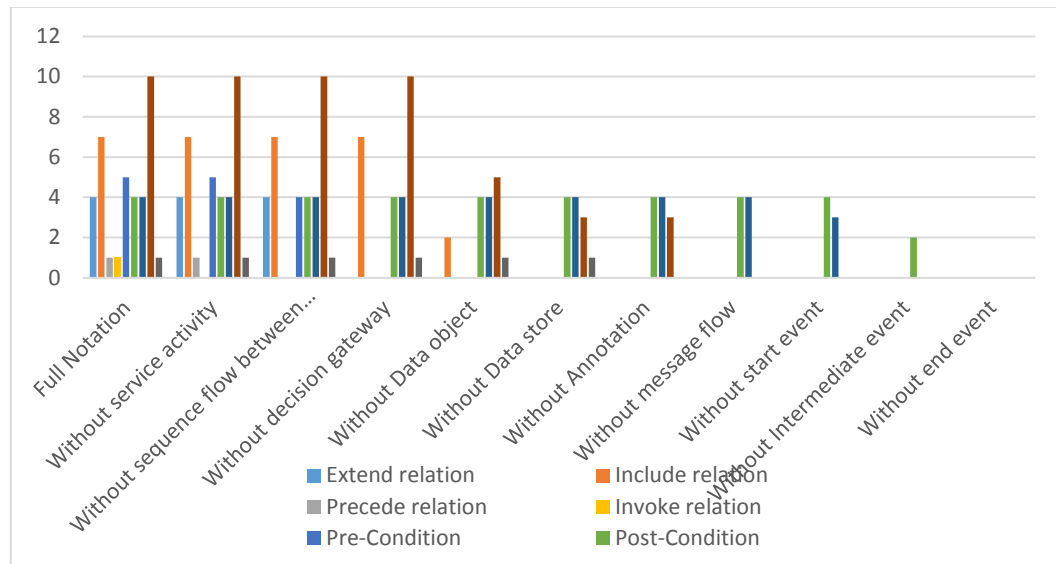


Figure 19: the effect of model richness on the efficiency of BMSpec.

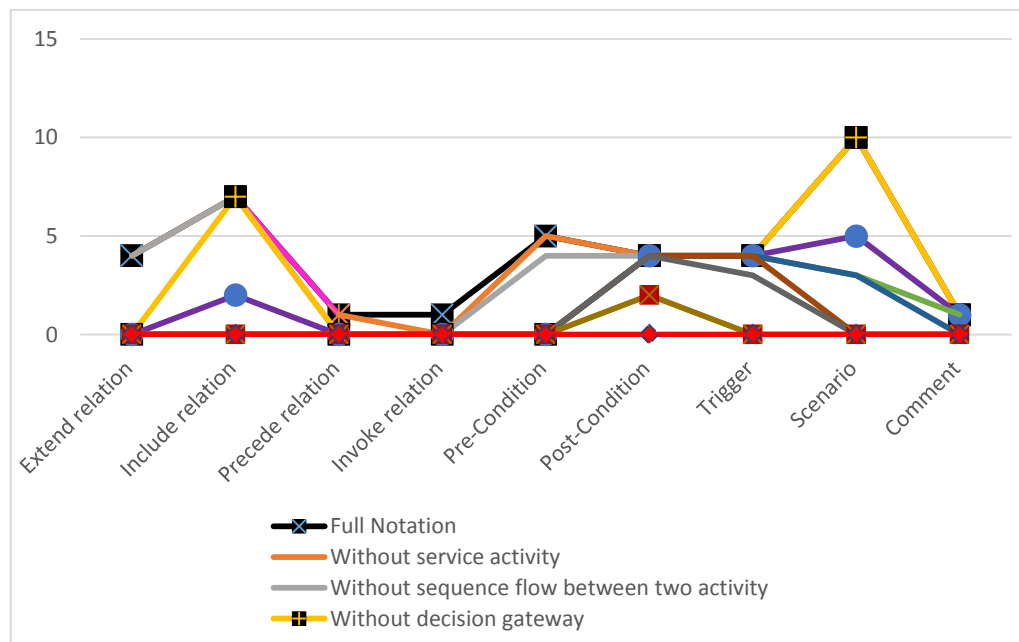


Figure 20: the effect of model richness on the efficiency of BMSpec.

Comparison between effects of level of richness on the efficiency of BMSpec

Now we will compare the efficiency of BMSpec regarding different models with different richness levels. As we mention in the evaluation methodology, there are six levels of richness and thirty business process models distributed on these levels, the results are shown in Table 22. Table 23 shows the time required to transform models to a well-defined model.

Model Name	Input Business process model Notation															Output use case diagram element					Output use case description element									
	Activity Num.	Performet Num.	Sequence Flow between activity and Service	Sequence Flow between two activities Num.	Decision Gateway bet, two activities Num.	Input Data object Num.	Output Data object Num.	Input Data store Num.	Output Data store Num.	Annotation Num.	Input message flow between activities Num.	Output me	Start Event Num.	Intermediate Event (None) Num.	Intermediate Event (Timer) Num.	Intermediate Event (Message) Num.	End event Num.	Use Case Num.	Actors Num.	Extend relation Num.	Include relation Num.	Precede relation Num.	Invoke relation Num.	Use Case Desc. Num.	Use Case Pre-Condition Num.	Use Case Ppst-Condition Num.	Use Case Trigger Num.	Use Case Scenario sentence Num.	Use Case Comment Num.	
LOR1: of richness																														
Open Vacancy	8	3	1	2	2	2	3	1	1	0	0	1	1	1	0	0	2	15	3	2	7	2	1	8	5	2	2	8	0	
Invoice receipt	5	3	1	1	2	0	0	1	0	0	0	0	1	0	0	0	2	5	3	2	1	1	1	5	5	1	1	1	0	
LOR2: of richness																														
Purchase via email	10	2	0	3	3	2	3	0	0	0	2	2	1	0	0	0	2	15	2	3	5	3	0	10	6	1	1	9	0	
Order to cash	8	3	0	3	1	2	2	0	0	0	1	1	1	0	0	1	4	12	3	1	4	3	0	8	4	4	4	6	0	
Auction	6	2	0	1	1	1	1	0	0	0	5	5	1	0	0	1	1	8	2	1	2	1	0	6	3	2	2	12	0	
Water purchase process	6	4	0	2	1	2	2	0	0	1	1	1	1	0	1	0	1	10	4	1	4	2	0	6	3	2	2	6	1	
Travel agency	4	3	0	1	2	2	2	0	0	0	3	3	1	0	0	0	1	8	3	2	4	1	0	4	3	1	1	10	0	
Request to proposal	6	2	0	2	3	0	0	1	1	0	0	0	2	0	0	0	4	8	2	3	2	2	0	6	5	0	2	2	0	
Claim assignment	5	1	0	2	1	8	0	2	0	0	0	0	1	0	0	0	2	15	1	1	10	2	0	5	4	1	1	10	0	
Collect parts order	8	2	0	1	3	4	3	0	0	0	1	2	1	0	1	0	2	15	3	3	7	1	0	8	4	4	4	10	0	
LOR3: of richness																														
Cab booking process	8	3	1	1	3	0	0	0	0	0	4	4	1	0	1	1	2	8	3	3	0	1	1	8	5	2	2	8	0	
book-selling-process	7	5	0	1	1	0	0	0	0	0	4	2	1	0	0	1	1	7	5	1	0	1	0	7	3	2	2	6	0	
Customer pricing scenario	8	2	0	3	1	0	0	0	0	0	0	4	1	4	0	0	4	8	2	1	0	3	0	8	7	4	1	4	0	
Leather	8	2	0	2	2	0	0	0	0	0	4	4	1	0	0	0	1	8	2	2	0	2	0	8	6	1	1	8	0	

- ❖ Number of precede relation= number of sequence flow between two activities.
- ❖ Number of precondition= number of number of sequence flow between gateway and activity + number of sequence flow between activities.
- ❖ Number of scenario sentence = number of data association and message flow.

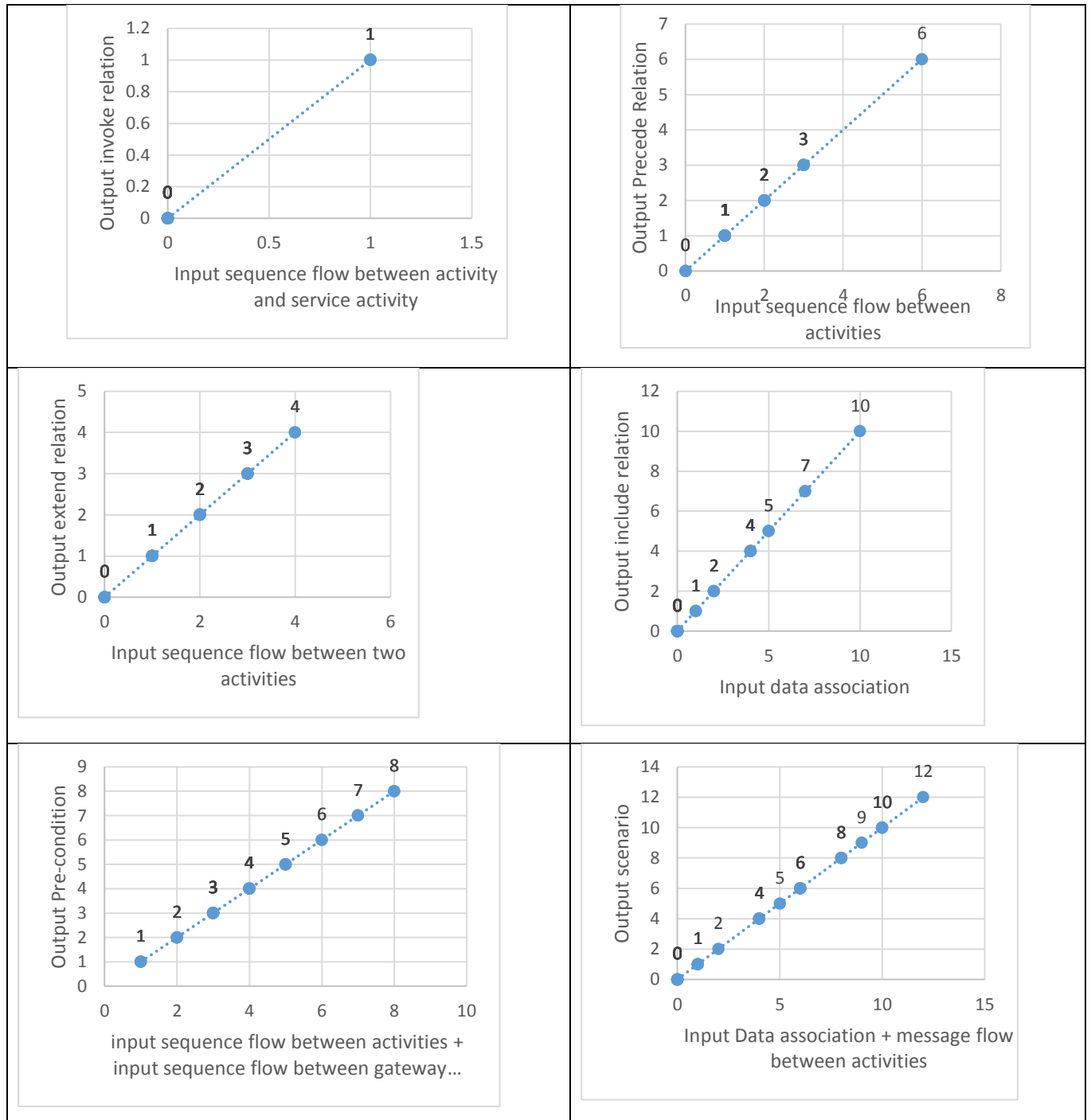


Table 23: Effect of number of input notation on the efficiency of BMSpec.

2. Type of notation in the input business process model

Type of notation can affect the efficiency of BMSpec as shown in different level of richness:

LOR1 contains two business process models, each model has the covered notations: service task, data association, decision gateway, events, message flow, and sequence flow between two activities. The generated use case diagram contains four types of association between use cases: include, extend, invoke, and precede and the use case description contains all items precondition, post-condition, triggers, and scenario.

LOR2 contains eight business process models, each model has the covered notations except service task type of activity. This is clear in the generated use case diagram it contains three types of association between use cases: include, extend, and precede, the invoke association disappeared, But the use case description does not effected with the missing service task, it contains precondition, post-condition, triggers, and scenario.

LOR3 contains five business process models, each model does not cover data association such as: input data object, output data object, input data store and output data store. This will affect the generated use case diagram element as shown in Figure 21, it contains precede and extend association, without include association between use cases also the scenario in the generated use case description will be reduced.

LOR4 contains four business process models, each model does not covered decision gateway. This will affect the generated use case diagram element as shown in Figure 21, it contains precede and include, without extend association between use cases also the precondition in the generated use case description will be reduced.

LOR5 contains five business process models, each model does not cover data association such as: input data object, output data object, input data store and output data store. Also it does not cover the message flow notation. This will affect the generated use case diagram element as shown in Figure 21, no include association between use cases also scenario in the generated use case description because the scenario generated only from data association and message flow.

LOR6 contains six business process models, this level is the lowest richness level, as we can notice from Figure 21, it is not homogeneous notation missing. Some of models not covered data association and decision gateway at the same time which can affect the include association, extend association, precondition and scenario. Some other models not covered precede association and service association at the same time, some other models not covered precede, invoke and message flow.

We can use this classification when the business model not fit into levels: LOR 1, LOR 2, LOR 3, LOR 4, and LOR 5.

Comparison between the effects of model richness on the efficiency of BMSpec.

We have six levels of richness, from Figure 21 we can compare the effect of richness on the efficiency of BMSpec:

Comparing LOR1 and LOR 2 we can notice the impact of eliminating service task, the invoke association reduced to zero.

Comparing LOR1, LOR2 and LOR3 we can notice the impact of eliminating data association, the include association between use case is reduced to zero.

Comparing LOR1, LOR2, LOR 3 and LOR 4 we notice the impact of elimination decision gateway notation on the efficiency, the extend association

reduced to zero also the pre-condition reduced but not eliminated to zero because the event play a main role in generating precondition.

Comparing LOR1, LOR2, LOR3, LOR4 and LOR5 we can notice the impact of eliminating data association and message flow, the include association between use case is reduced to zero also scenario in use case description reduced to zero because the scenario generated only from data association and message flow.

Comparing LOR1, LOR2, LOR3, LOR4, LOR5 and LOR6 we can notice the impact of eliminating sequence flow between two activities, the precede association between use case is reduced also this will implicitly eliminate invoke association. Scenario in use case description will be reduced because of eliminating message flow, but not reduced to zero because the scenario generated also from data association.

Also eliminating decision gateway and data association at the same time will result in eliminating include and extend association, scenario and precondition.

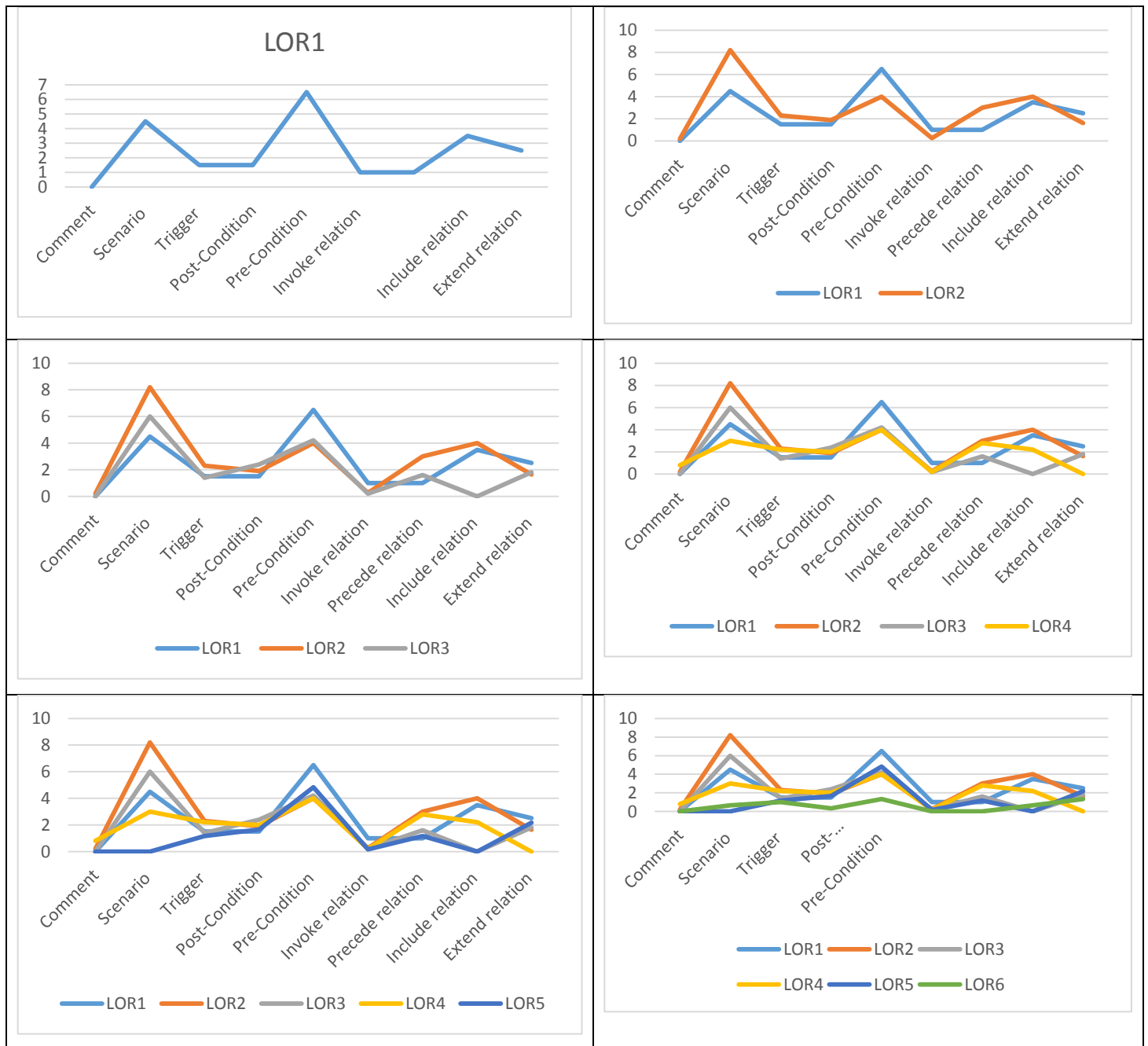


Figure 21: comparison between different levels of richness.

The time required to transform each business process model to a well-defined model is listed in Table 24, these models need a simple clarification and the time required to achieve it measured in minutes.

Model Name	Type of Transformation	Time In Hours.
Loan request	1. Declare start and end event 2. Declare decision gateway condition	1 minute
Leather manufacturing	1. Declare start and end event 2. Clarify the splitting and decision gateway in to two gateway 3. Declare message flow	2 minute
Authoring process	1. Message flow from event to activity	1 minute
Class registration	1. Declare start and end event	1 minute
Leave request	1. Declaration decision gateway	1 minute
Cab booking process	1. Declare timer event 2. Declare message flow	1 minute
Travel Plan Quote Process	1. Declare start event 2. Declare inclusive condition and complex condition	2 minute
Claim assignment	1. Declare decision condition	1 minute
Credit card issuance	1. Declare start event	1 minute
manufacturing	1. Declare start and end event.	1 minute
Customer pricing scenario	1. Declare gate way condition 2. Declare end event	2 minute
license lifecycle App.	1. Declare start and end event	1 minute
Purchase via email	1. Declare end event 2. Declare decision gateway condition	2 minute
Collect parts order	1. Declare start and end event	1 minute
Water purchase	1. Declare start event and end event 2. Declare message flow name.	2 minute
Collect vote	1. Declare start event	1 minute

Table 24: time and type of model transformation.

6.2 Correctness evaluation results

After evaluating the efficiency of BMSpec, we will evaluate the correctness by using seven business models and compare the results of BMSpec with result of manual work. We start the evaluation by getting the existing business process model for these processes, then transformed them into a well-defined business model, the transformation type depends on the status of the existing business process model as we mention previously in BMSpec steps. The existing business model needs to re-engineering and defining the system functionality. The time required for this transformation and type of transformation is listed in Table 26.

While the time required to write requirement specification using traditional requirement engineering techniques is listed in Table 27.

Model Name	Type of Transformation	Time In Hours.
registration on X-Road	Re-engineering transformation: The existing business process model consist of five tasks distributed in to two pools and send messages between the two pools. After re-engineering consists of two tasks distributed in to two pools while three manual tasks were deleted and a new gateway added.	2 hours.
Consume X-Road service	Re-engineering transformation: The existing business process model consist of twelve tasks distributed in to thee pools and send messages between pools, some of these task are manual tasks, After re-engineering consists of four tasks distributed in to thee pools and send messages between pools, two new condition gateway added to determine the sequence and flow of tasks.	3 hours
provide X-Road service	Re-engineering transformation: The existing business process model consist of six tasks distributed in to two pools and send messages between the two pools, After re-engineering consists of three tasks distributed in to two pools and send messages between the two pools, with new condition gateway added.	3 hours
Car Hire case study(hire car process)	the business process model is ready no need to transformation.	-
Online Bookshop System(make order process)	the business process model is ready no need to transformation	-
Extra feature calculator	the business process model is ready no need to transformation	-
Nobel Prize example	the business process model is ready no need to transformation	-

Table 25: time and type of business process model transformation using BMSpec.

Case study Name	Work in hors
registration on X-Road	It is a real work done in Palestinian ministry of telecom and IT. Jul- 2015. All work hour documented. Two expert work on it. 2 hours per day 3 days in week. 1 week. Total working person hour= $2*2*3= 12$ hour.
Consume X-Road service	It is a real work done in Palestinian ministry of telecom and IT. Jul- 2015. All work hour documented. Two expert work on it. 2 hours per day 4 days in week. 1 week. Total working person hour= $2*2*4=16$ hour.
provide X-Road service	It is a real work done in Palestinian ministry of telecom and IT. Jul- 2015. All work hour documented. Two expert work on it. 2 hours per day 4 days in week. 1 week. Total working person hour= $2*2*4= 16$ hour.
Car Hire case study(hire car process)	It is a study done in web engineering course spring -2015. All meetings minute exist. Four expert student works on it. 2 hours per week meeting. Along 10 weeks. Total working person hour= $2*4*10= 80$ hour.
Online Bookshop System(make order process)	It is a study done in web engineering course spring -2015. All meetings minute exist. Four expert student works on it. 4 hours per week meeting. Along 10 weeks. Total working person hour= $4*4*10= 160$ hour.
Extra feature calculator	It is a study done in software engineering course 2014. All meetings minute exist. Four expert student works on it. 2 hours per week meeting. Along 4 weeks. Total working person hour= $2*4*4= 32$ hour.

Table 26: time and type of business process model transformation using manual work.

The comparison between the time required to generate requirement specification between BMSpec and manual work using traditional requirement engineering techniques are shown in Table 27.

Case study Name	Time required in manual work	Time required in BMSpec
registration on X-Road	12 hours	2 hours
Consume X-Road service	16 hours	3 hours
provide X-Road service	16 hours	3 hours
Car Hire case study(hire car process)	24 hours	3 minutes
Online Bookshop System(make order process)	30 hours	3 minutes
Extra feature calculator	20 hours	3 minutes

Table 27: comparison between time required for manual work and BMSpec

In **Car Hire case** study they spend 80 person working hour to generate 10 use cases, so the average time for each use case is 8 hours.

We compare the required time to generate 3 use case. So the total time required in manual work to generate 3 use cases diagram and description = 24 hours. While in BMSpec we spend only minutes to run the algorithm and get the results

In **online book shop** study they spend 160 person working hour to generate 32 use cases, so the average time for each use case is 5 hours.

We compare the required time to generate 6 use cases. So the total time required in manual work to generate 6 use cases diagram and description = 30 hours. While in BMSpec we spend only minutes to run the algorithm and get the results

In **Extra feature calculator** they spend 32 person working hour to generate 3 use cases, so the average time for each use case is approximately 10 hours We compare the required time to generate 2 use cases. So the total time required in

manual work to generate 6 use cases diagram and description = 20 hours. While in BMSpec we spend only minutes to run the algorithm and get the results.

The result of Applying BMSpec on these seven business process models are listed in Table 28. And comparison between result of BMSpec and manual work are listed in Table 29, in order to measure the correctness percentage accurately, we compare the quantity and validity, the extra features in BMSpec are listed but not included in the calculation of correctness percentage.

Model Name	Input Business process model Notation																Output use case diagram element				Output use case description element								
	Activity Num.	Performer Num.	Service activity Num.	Sequence Flow between two activities Num.	Exclusive Decision Gateway Num.	Input Data object Num.	Output Data object Num.	Input Data store Num.	Output Data store Num.	Annotation Num.	Input message flow Num.	Output message flow Num.	Start Event Num.	Intermediate Event (None) Num.	Intermediate Event (Timer) Num.	Intermediate Event (Message) Num.	End event Num.	Use Case Num.	Actors Num.	Extend relation Num.	Include relation Num.	Precede relation Num.	Invoke relation Num.	Use Case Desc. Num.	Use Case Pre-Condition Num.	Use Case Post-Condition Num.	Use Case Trigger Num.	Use Case Scenario sentence Num.	Use Case Comment Num.
registration on X-Road	4	2	0	0	1	1	1	0	0	0	1	1	2	0	0	0	2	6	2	1	2	0	0	4	1	1	2	4	0
Consume X-Road service	7	3	0	1	2	2	1	0	0	0	2	5	2	0	0	0	2	10	3	3	3	1	0	7	4	1	2	10	0
provide X-Road service	5	2	0	1	1	1	1	0	0	0	2	1	2	0	0	0	2	7	2	1	2	1	0	5	2	1	2	5	0
Car Hire case study(hire car process)	3	2	0	2	0	1	2	0	0	1	6	3	1	2	0	0	1	3	2	0	3	2	0	3	2	1	3	12	1
Online Bookshop System(make order process)	6	3	0	2	1	1	1	1	2	0	13	7	1	5	0	0	1	6	3	2	5	2	0	6	6	1	6	25	0
Extra feature calculator	2	1	0	0	0	4	1	0	0	0	0	0	2	0	0	1	1	7	1	0	1	0	0	2	0	1	3	5	0
Nobel Prize example	10	4	0	7	1	1	1	4	3	1	2	3	1	0	0	0	1	19	4	1	9	7	0	10	9	1	1	14	1

Table 28: result of running the proposed algorithm on seven business process model.

	Manual work		BMSpec		Correctness Percentage	Extra feature BMSpec
Nobel Prize example						
	Qty.	valid	Qty.	valid		
Use case.	10	10	10	10	100%	Include association Extend association Precede association Invoke association Comment.
Actor	4	4	4	4	100%	
Use case Desc.	3	3	3	3	100%	
Trigger	1	1	1	1	100%	
Pre-condition	2		2	2	100%	
scenario	6	6	6	6	100%	
Average					100%	
Extra feature calculator						
	Qty.	valid	Qty.	valid		
Use case.	2	2	2	2	100%	Post-condition
Actor	1	1	1	1	100%	
Use case Desc.	2	2	2	2	100%	
Trigger	2	2	2	2	100%	
Pre-condition	2(redundant)	0	0 exist in scenario	0		
scenario	9 (redundant)	5	5	5	100%	
Invoke association	1	1	0 covered as trigger and post condition	1	100%	
Average					100%	
registration on X-Road						
	Qty.	valid	Qty.	valid		
Use case.	4	4	4	4	100%	
Actor	2	2	2	2	100%	
Use case Desc.	4	4	4	4	100%	
Trigger	2	2	2	2	100%	
Pre-condition	2	1	1	1	100%	
scenario	4	3	3	3	100%	
extend association	1	1	1	1	100%	
Average					100%	
Consume X-Road service						
	Qty.	valid	Qty.	valid		
Use case.	7	7	7	7	100%	
Actor	3	3	3	3	100%	
Use case Desc.	7	7	7	7	100%	

Trigger	2	2	2	2	100%	
Pre-condition	5(1 redundant)	4	4	4	100%	
scenario	8(1 redundant)	7	7	7	100%	
extend association	3	3	3	3	100%	
Average					100%	
provide X-Road service						
	Qty.	valid	Qty.	valid		
Use case.	5	5	5	5	100%	
Actor	2	2	2	2	100%	
Use case Desc.	5	5	5	5	100%	
Trigger	2	2	2	2	100%	
Pre-condition	3(1 redundant)	2	2	2	100%	
scenario	6(1 redundant)	5	5	5	100%	
extend association	1	1	1	1	100%	
Average					100%	
Car Hire case study(hire car process)						
	Qty.	valid	Qty.	valid		
Use case.	3	3	3	3	100%	comment precede association
Actor	2	2	2	2	100%	
Use case Desc.	3	3	3	3	100%	
Trigger	3	3	3	3	100%	
Pre-condition	3(1 redundant)	2	2	2	100%	
scenario	14(2 redundant)	12	12	12	100%	
Average					100%	
Online Bookshop System (make order process)						
	Qty.	valid	Qty.	valid		
Use case.	5	5	5	5	100%	Extend association precede association include association
Actor	3	3	3	3	100%	
Use case Desc.	5	5	5	5	100%	
Trigger	6	6	6	6	100%	
Pre-condition	6	6	6	6	100%	
scenario	33(redundant)	25	25	25	100%	
Average					100%	

Table 29: comparison between result of BMSpec and manual work with correctness ratio

Nobel Prize example:

According to work in paper [37] the result use case diagram consists of ten use cases without any association as shown in figure22, because in their approach they don't focus or handle the associations between use cases.

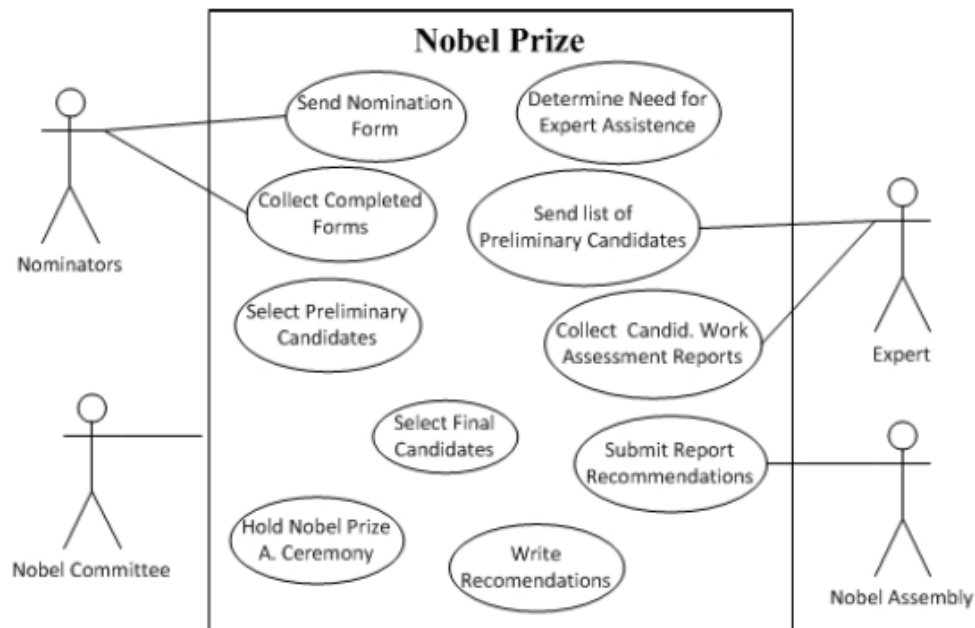


Figure 22: The Nobel Prize Use Case Diagram [31].

The generated use case diagram from BMSpec shown in figure 23, it includes extra four types of associations between use cases (include, precede and extend).

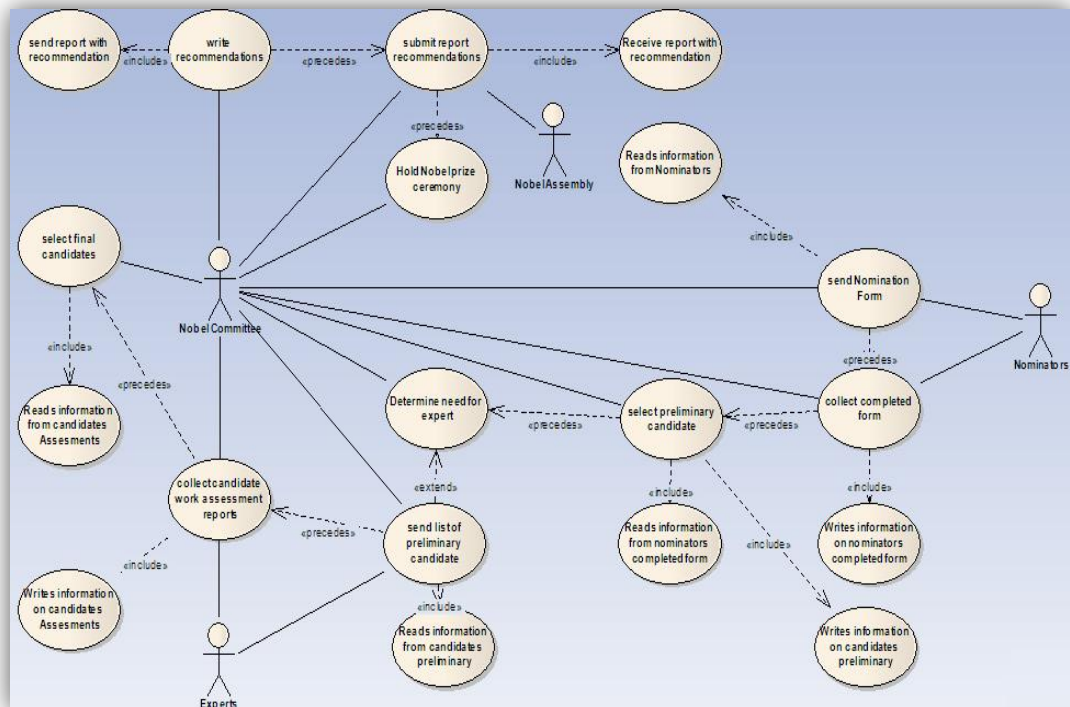


Figure 23: The Nobel Prize Use Case Diagram From BMSpec

In [31] they present only three use case description from the ten use case description as shown in figure24, figure25 and figure26, the same use case description generated from BMSpec are shown in table31, table32 and table33.

Use Case name	Send Nomination Form.
Actors	Nobel Committee, Nominators
Trigger	The time-date September is reached.
Scenario	Around 3000 invitations confidential nomination forms are sent to selected Nominators. Reads information from Nominators. Sends the Nomination Invitation to Nominators.

Figure 24: Send Nomination Form use case description[31]

Use Case Name	send Nomination Form
Actors	Nominators , Nobel Committee
Trigger	The time-date September is reached.
Pre-Condition	
Scenario	Reads information from Nominators. Send Nomination invitation To Nominators
Comment	Around 3000 invitation confidential nomination form are send to selected nominators

Table 30: Send Nomination Form use case description From BMSpec

Use Case name	Send List of Preliminary Candidates.
Actors	Nobel Committee, Expert
Pre-condition	The Expert Assistance Required? is Yes.
Scenario	Reads information from Preliminary Candidates. Sends the Candidates to be Assessed to Expert.

Figure 25: Send List of Preliminary Candidates use case description [31]

Use Case Name	send list of preliminary candidate
Actors	Nobel Committee , Experts
Trigger	
Pre-Condition	The expert assistance required? is Yes
Scenario	Reads information from candidates preliminary. Send Candidates to be assessed To Experts

Table 31: Send List of Preliminary Candidates use case description from BMSpec

Use Case name	Write Recommendations.
Actors	Nobel Committee
Pre-condition	The Expert Assistance Required? is No or Select Final Candidates has been completed.
Scenario	Sends The Report with Recommendations.

Figure 26: Write Recommendations use case description[31]

Use Case Name	write recommendations
Actors	Nobel Committee
Trigger	
Pre-Condition	The expert assistance required? has been completed. exclusive or select final candidates has been completed.
Scenario	Send report with recommendation.

Table 32: Write Recommendations use case description From BMSpec

Extra feature calculator:

According to manual work in extra feature calculator the result use case diagram consists of two use cases with one invoke association, this invoke association does not appear in our results because we cover the association between these two activities as a trigger and not invocation the do money exchange activity send a trigger to do simple calculation activity.

In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, a second difference is in dealing with pre-condition, in manual work some of scenario sentences appear also in the precondition, while in BMSpec it appear only in scenario, a third difference in manual work they deal sometimes with post-condition as a step in scenario while in BMSpec we deal with it as only post-condition. So we can notice different in number of sentences in scenario and precondition between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it. There is not any information loss. The correctness percentage is 100% with Post-condition extra feature, detailed results in ii. Appendix.

Registration on X-Road:

According to manual work in registration on X-road process the result use case diagram consists of four use cases with one extend association the same result we get from BMSpec. In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, Another difference is in dealing with pre-condition, in manual work some of scenario sentences appear also in the precondition, while in BMSpec it appears only in scenario. So we can notice different in number of sentences in scenario and precondition between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it. There is not

any information loss. The correctness percentage is 100%, detailed results in ii. Appendix.

Consume X-Road service:

According to manual work in Consume X-road service process the result use case diagram consists of seven use cases with three extend association.

In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, Another difference is in dealing with pre-condition, in manual work some of scenario sentences appear also in the precondition, while in BMSpec it appears only in scenario. So we can notice different in number of sentences in scenario and precondition between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it. There is not any information loss. The correctness percentage is 100%, detailed results in ii. Appendix.

Provide X-Road service:

According to manual work in Consume X-road service process the result use case diagram consists of seven use cases with three extend association the same result we get from BMSpec.

In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, Another difference is in dealing with pre-condition, in manual work some of scenario sentences appear also in the precondition, while in BMSpec it appears only in scenario. So we can notice different in number of sentences in scenario and precondition between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it. There is not any information loss. The correctness percentage is 100%, detailed results in ii. Appendix.

Car Hire case study (hire car process):

According to manual work in hire car process the result use case diagram consists of three use cases with two actors, the same result we get from BMSpec as shown in table33, but BMSpec has extra features: Precede association

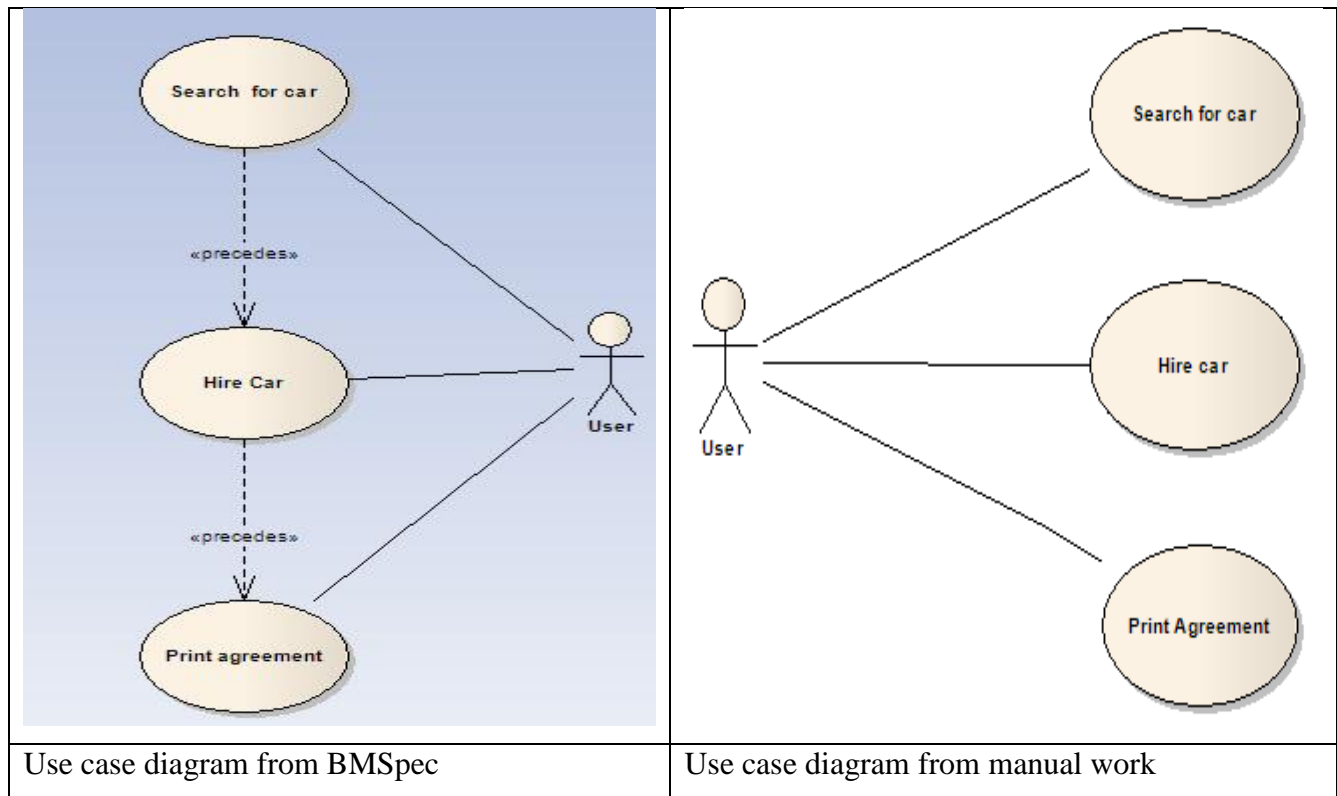


Table 33: comparison between use case diagram generated from manual work and BMSpec

In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, Another difference is in dealing with pre-condition, in manual work some of scenario sentences appear also in the precondition, while in BMSpec it appear only in scenario. So we can notice different in number of sentences in scenario and precondition between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it, generated use case description from manual work represented in figure27 and figure 28 , while generated use case description from BMSpec represented in table34 and table35. There is not any information loss. The correctness percentage is 100%.

Title	Search for car
Actors	Guest, user, car hire agency, administrator
Preconditions	The user is logged in.
Sequence/Flow of Events	<ol style="list-style-type: none">1. The system display the search form for the user.2. The user enter the criteria to search for the cars.3. The system display a list of matched cars based on the entered criteria.
Stimulus/Trigger	The user click search button

Figure 27: Search for car use case description from manual work

Use Case Name	Search for car
Actors	user, car hire agency, administrator, Guest
Trigger	The Event click search button occurred
Pre-Condition	
Scenario	Receive Search form from system. Send search criteria To System. Receive list of matched cars from System.

Table 34: Search for car use case description from BMSpec

Title	Hire car
Actors	User, car hire agency, administrator
Preconditions	The user searched for car.
Sequence/Flow of Events	<ol style="list-style-type: none">1. The user search for the car that he wants.2. The user select the car that he want to hire from the search results, and click the hire button.3. The system display the hire car form.4. The user fill the hire form.5. The system send payment form.6. The user send payment method.7. The system display payment notification.
Stimulus/Trigger	Click the hire button of the car.

Figure 28: Hire car use case description from manual wor

Use Case Name	Hire Car
Actors	user, car hire agency, administrator
Trigger	The event click hire button occurs.
Pre-Condition	The Search for car has been completed.
Scenario	Receive hire form from System. Send hire data to System. Receive payment method form from System. Send payment method To System Receive payment notification from System

Table 35: Hire car use case description from BMSpec

Online Bookshop System (make order process):

According to manual work in Consume X-road service process the result use case diagram consists of five use cases with three actors, six triggers and six pre-condition the same result we get from BMSpec.

In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, So we can notice different in number of sentences in scenario between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it. There is not any information loss. The correctness percentage is 100% with extra features: Extend association, precede association and include association, detailed results in ii. Appendix..

6.3 Sufficiency evaluation results

Now we will evaluate the sufficiency of BMSpec, Table 38 shows the covered relations between notations in each business process model and uncovered notation, also the percentage of coverage is shown in Figure 29.

Model Name	Covered	Uncovered	Uncovered notation type
Insurance company offer	9	0	
Hardware retailer	9	2	Sequence flow between two gateways.
Loan request	7	1	sequence flow between gateway and event
Employee assignment	4	1	sequence flow between gateway and event
Leather manufacturing	12	1	Sequence flow between two gateway
Employee expenses approval	9	3	<ul style="list-style-type: none"> • Sequence flow between two gateway • sequence flow between gateway and event
Authoring process	10	0	
Class registration	6	3	sequence flow between gateway and event
Leave request	5	2	<ul style="list-style-type: none"> • Sequence flow between two gateways. • Sequence flow between gateway and event.
Online shop	7	1	<ul style="list-style-type: none"> • Sequence flow between two gateways.
Cab booking process	14	2	<ul style="list-style-type: none"> • Sequence flow between two event • Sequence flow between gateway and event.
Travel Plan Quote Process	7	3	<ul style="list-style-type: none"> • Sequence flow between two event • Sequence flow between gateway and event.
Claim assignment	15	2	<ul style="list-style-type: none"> • Sequence flow between two gateways. • Sequence flow between gateway and event.
Credit card issuance	7	1	Sequence flow between gateway and event.

Invoice receipt	8	1	Sequence flow between gateway and event.
manufacturing	12	0	
Customer pricing scenario	16	6	<ul style="list-style-type: none"> • Sequence flow between two gateways. • Sequence flow between two event
license lifecycle App.	22	1	Sequence flow between gateway and event.
Request to proposal	9	5	Sequence flow between gateway and event.
Book selling process	17	2	Sequence flow between two gateways.
Open Vacancy	20	1	Sequence flow between gateway and event.
Purchase via email	15	1	Sequence flow between gateway and event.
Order to cash	17	1	Sequence flow between gateway and event.
Auction	14	1	Sequence flow between two gateways.
Collect parts order	20	0	
Water purchase	13	1	Sequence flow between gateway and event
Travel agency	15	0	
Account creation	4	1	Sequence flow between gateway and end event.
Working group	3	2	<ul style="list-style-type: none"> • Sequence flow between gateway and event. • Sequence flow between two event
Collect vote	13	3	<ul style="list-style-type: none"> • Sequence flow between two events • Sequence flow between gateway and event. • Sequence flow between gateway and event.
Total	339	48	

Table 36: the result of sufficiency evaluation: covered and uncovered notation

From Table 38 we can notice that there are 3 types of notation is not covered in BMSpec:

- **Sequence flow between two events**

In some cases we find a sequence flow connects two events, the relation between these events are not covered in BMSpec. But the sequence flow between events and activities is covered, so we are not sure if there are some useful data that we can extract from this sequence flow between events, we are planning to evaluate this in the future work.

- **Sequence flow between two gateways.**

In some cases we find a sequence flow connects two gateways, the relation between these gateways are not covered in BMSpec. The gateways in business process models affects the executing of the workflow of business, so we think there is some data exists in this notation between two gateways that we can extract and transformed it to precondition in the generate use case description, this work may be done in future.

- **Sequence flow between gateway and end event.**

In some cases we find a sequence flow connects between gateway and event, the relation between these notations are not covered in BMSpec. The gateways in business process models affects the executing of the workflow of business, and the event also, determine when process will be ended or executed so we think there is some data exists in this notation between gateway and event that we can extract and transformed it to post-condition or precondition in the generate use case description, this work may be done in future.

BMSpec can cover 27 types of relations, while 3 types are not covered which means that the sufficiency percentage reaches to 90% shown in Figure 29/

The total covered relation in the thirty business model is 339 and the uncovered is 48 which means the usage of covered relations is 87.6%, while the uncovered relations is 12.4% as shown in Figure 30, from these results we can conclude the uncovered notation is not more popular or usable more than the covered notation. Also the benefit of covering them is not surely significant it need investigation, so

we postpone the working on it in the future work. BMSpec is sufficient enough to cover most of business process notations and relations, so it extracts most valuable data to build an enough meaning full use case model.

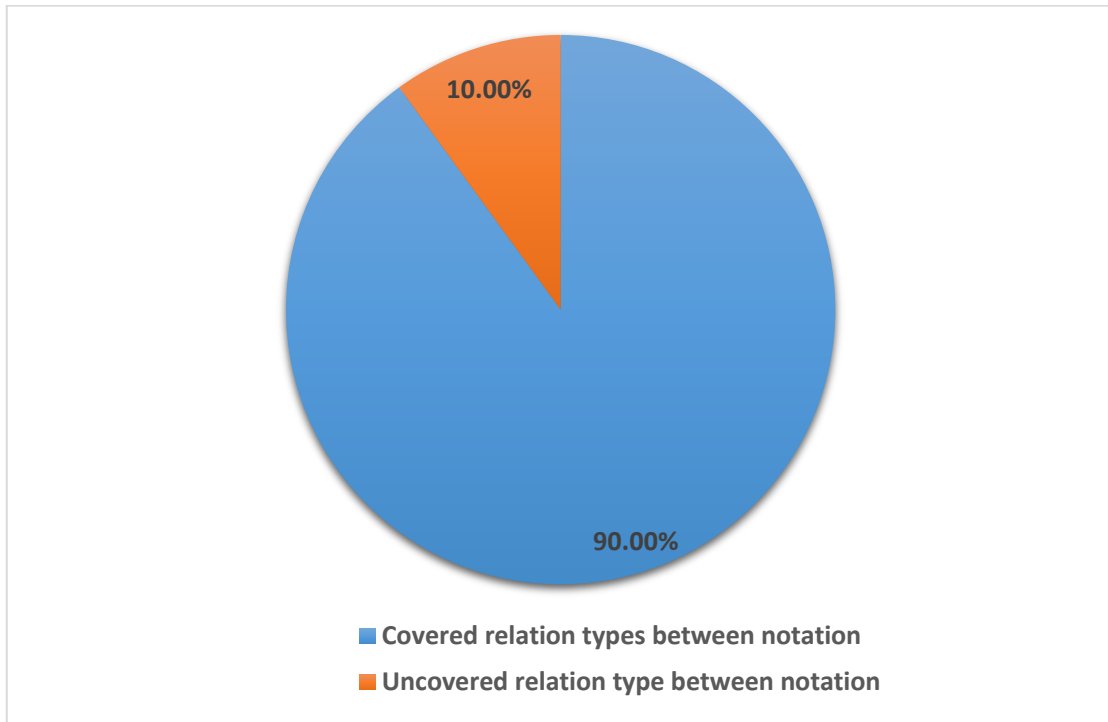


Figure 29: sufficiency of proposed approach

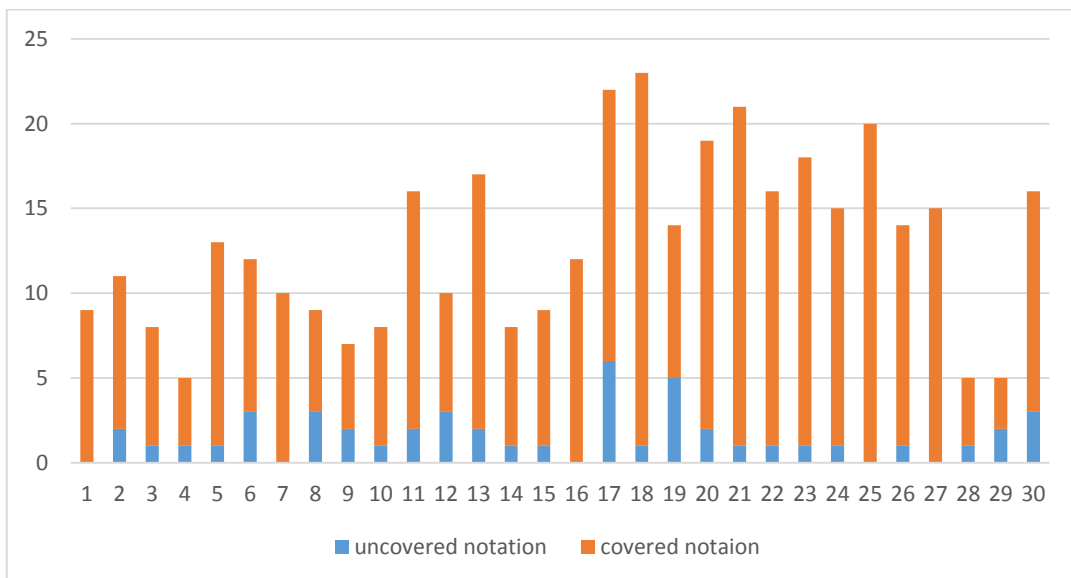


Figure 30 : usage covered relation between notations

Chapter 7

Conclusion

This chapter concludes the thesis. A summary of literature review is presented With focus on the main contributions, results, limitations and assumptions, and Future work.

7.1 Introduction

In our literature review, we focused on business process based approaches to get requirement specification. Because we have shown that the generated requirement specification is accurate and can be generated faster compared to traditional requirement engineering techniques. In the existing literature, there are many approaches to generate use case diagram from business process model [1, 2, 6, 31, 7], some of them is automated algorithm and some of them just a general rule for manual transformation. The main drawback of the existing automated algorithm is that absence of association between use cases. Also, they transform business model AS_Is without defining the software system goals, so many manual task will be generated and redundant use case, which means the result use case diagram is not efficient and need to be correct manually by human. Another drawback of these approaches is the output just a use case diagram without use case description [1, 2, 6, 7, 32].

Chapter 4 proposed a new business process based approach which can generate automatically a rich use case model with four types of association (include, invoke, extend and precede) also it can generate a rich use case description for each use case in a very short time compared to the other approaches, table 37 shows how BMSpec addresses the limitations in the related existing work. To the best of our knowledge, it is the first approach that can automatically transform business process models to Use Case models that include generated detailed use

case descriptions and associations between use cases within minimal manual intervention.

Approach	Automated Transformation	Output		
		Use case diagram	Association between use cases	Use case Description.
Dijkman et al, 2002 [1]	+	+	-	-
Odeh et al, 2003 [2]	+	+	-	-
Aburub et al, 2012 [6]	+	+	-	-
Cruz et al, 2014 [31]	-	+	-	+
Adam et al, 2014 [7]	-	+	+	-
Rhazali et al, 2014 [32]	-	+	+	-
BMSpec	+	+	+	+

Table 37: How BMSpec addresses limitaions in related existing work.

The rest of this chapter presents the conclusion, including: contributions, summary of results, limitation and assumptions. Also, it presents potential research areas for future work.

7.2 Contribution

The overall goal of this thesis it to improve requirement engineering process by getting more related to business need requirements in shorter time. To achieve this overall goal, we defined a systematic approach (BMSpec) to generate requirement specification, BMSpec consist of a set of steps summarized as below:

- Defined 9 rules to prepare a well-defined business process model using business process reengineering.
- Defined 2 main rules with 8 sub-rules to make business model simple clarification.
- Defined 10 Heuristic rules to map business process concepts to use case diagram concepts.

- Transformation algorithm has been developed, this algorithm can automatically transform business process models to UML use case models. Based on a predefined 10 heuristic rules.
- Defined 26 Heuristic rules to map business process concepts to use case description details.
- Transformation algorithm has been developed, this algorithm can automatically transform business process models to use case description details. Based on a predefined 26 heuristic rules.

7.3 Results

We use three type of evaluation to evaluate the correctness, sufficiency and efficiency of BMSpec. The results summarized as bellow:

- **Correctness evaluation**, we compare BMSpec with 7 case studies 6 of these cases use traditional requirement engineering elicitation techniques to get requirement specification, and 1 case uses manual transformation from business process model to requirement specification. Results show our work achieved extra features which is not exist in other approaches such as association between use cases (precede , invoke, include , extend), also despite our work is automatically transform business model to requirement specification no loss of any information found, but there is some redundant information in the requirement specification using traditional requirement engineering techniques, such as defining precondition sentence both in scenario and precondition, also trigger sentence in scenario and trigger.

We compare the time required to generate requirement specification using manual work and BMSpec, we found a big difference, which show BMSpec achieve same results with extra features in a very short time and less effort.

- **Efficiency evaluation** we use different 30 business process model to show how the richness of input business model can affect the efficiency of BMSpec, we found that the effecincy of developed approach depends on two factors: number of notation and type of notation in the input business process model. The effecincy of developed approach works efficiently depending on the

number of notation as the number of notation increases the richness of the output use case model increases. Also the efficiency of developed approach works efficiently depending on type of notation as the type of notation increases the richness of the output use case model increases, we have six levels of richness, LOR1 is the highest richness on which BMSpec works most efficiently, then in each level we eliminate a specific type of notation and measure how this can affect the efficiency of BMSpec, we found when the level of richness of input business model decreases the efficiency of developed approach also decreases.

- **Sufficiency evaluation** we use different 30 business process models to show how much BMSpec is sufficient to cover relation between BPMN. We find 3 types of notation which we don't cover: sequence flow between two events, sequence flow between two gateways, sequence flow between event and gateway. But we are not sure if these notations can affect the result use case model. These notations need investigation, so, we put it in the future work. The total sufficiency percentage of developed approach is 90 %. At the same time these uncovered notations are frequently used the same as other covered notation, the percentage of usage of uncovered notation is 12.4% to the covered notation is 87.6% .

In this work we proposed a new systematic business model-driven approach for deriving UML-based requirement specifications, it consists of a set of systematic steps to derive use Case Model from business process model, the developed approach showed accurate results with higher generation efficiency directly proportional to the level of richness of the input business process model and high sufficiency of covering BPMN, but it's generating one of the UML-Based diagram (use case), so BMSpec does not cover or replace the entire requirements engineering stage, nor aims to generate comprehensive requirement specifications, however it provides a step forward to semi-automate the elicitation process through the extraction of as many as possible of requirements from existing underlying business process models. Thus it provides an aiding tool to business and system analysts in defining requirements, saving significant time and effort

toward reaching finalised requirements making the elicitation process more efficient.

7.4 Limitations and Assumptions

37 business process models with different notation and different level of richness were selected to evaluate our work for three types of evaluation. This section explains some limitations of our works as follow:

- Our work covered 90% of relation between BPMN. Three types of BPMN notation are not covered (Sequence flow between two events, Sequence flow between two gateways, and Sequence flow between gateway and end event.), which need further investigation on how they affect the output use case model.
- Some business rules and constraints may be specified as text the business model, it needs natural language processing techniques, which is outside the scope of our work.
- BMSpec focuses on generating the functional requirements, while the non-functional requirement are not covered.

7.5 Future Work:

This section presents the main areas for future work for our work as follows:

- We found three types of BPMN notation are not covered, so we intend to investigate the effect of these types on the output use case model.
- Extend BMSpec to automatically generate more UML-based models from business process model, such as UML activity diagram and UML Class diagram
- Extend BMSpec to be used in automated requirements traceability, so any changes in business process model will be reflected automatically on the requirement specifications (e.g. UML use case model, UML activity diagram and UML class diagram). This can solve the problem of volatility that arises in requirement elicitation phase, and allow the tracking of business growth and changes.

- Extend BMSpec to cover non-functional requirements.
- Extend BMSpec to automate the first step: preparing a well-defined business model, which currently is a manual step in BMSpec.

References

- [1] Dijkman, Remco M., S. M. Joosten, and Ordina Finance Utopics. "An algorithm to derive use case diagrams from business process models." Proc. of the 6th Intern. Conf. on Software Engineering and Applications (SEA), Anaheim, US. 2002.
- [2] Odeh, Mohammed, and Richard Kamm. "Bridging the gap between business models and system models." *Information and Software Technology* 45.15 (2003): 1053-1060.
- [3] Štolfa, Svatopluk, and Ivo Vondrák. "A description of business process modeling as a tool for definition of requirements specification." *Proceedings of System Integration 2004*. 2004.
- [4] de la Vara, Jose Luis, Juan Sánchez, and Óscar Pastor. "Business process modelling and purpose analysis for requirements analysis of information systems." *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 2008.
- [5] Bubenko, J., A. Persson, and J. Stirna. "D3 Appendix B: EKD User Guide." *Royal Institute of Technology (KTH) and Stockholm University, Stockholm, Sweden* (2001).
- [6] Aburub, Faisal. "Activity-based approach to derive system models from business process models." *Information Society (i-Society), 2012 International Conference on*. IEEE, 2012.
- [7] Przybylek, Adam. "A business-oriented approach to requirements elicitation." *Evaluation of Novel Approaches to Software Engineering (ENASE), 2014 International Conference on*. IEEE, 2014.
- [8] Harmon, P., Wolf, C.: *Business Process Modeling Survey*. Business Process Trends, 2011.
- [9] Bubenko, J. A., Persson, A., Stirna, J., *EKD User Guide*, Royal Institute of Technology (KTH) and Stockholm University, Stockholm, Sweden, 2001
- [10] Boehm, Barry W. *Software engineering economics*. Vol. 197. Englewood Cliffs (NJ): Prentice-hall, 1981.

-
- [11] White, Stephen A. "Introduction to BPMN." IBM Cooperation 2.0 (2004): 0.
 - [12] OMG, OMG Unified Modeling Language Specification version 1.4, OMG Specification formal/2001-09-67, 2001.
 - [13] Weerakkody, Vishanth, and Wendy Currie. "Integrating business process reengineering with information systems development: issues & implications." *Business process management*. Springer Berlin Heidelberg, 2003. 302-320.
 - [14] Hammer, Michael. "Reengineering work: don't automate, obliterate." *Harvard business review* 68.4 (1990): 104-112.
 - [15] Kaplan, R B & Murdock, L (1991), *Rethinking The Corporation: Core Process Redesign*, The Mckinsey Quarterly, November 2, 1991.
 - [16] White, Stephen A. "Introduction to BPMN." IBM Cooperation 2.0 (2004): 0.
 - [17] Monsalve, Carlos, Alain April, and Alain Abran. "On the expressiveness of business process modeling notations for software requirements elicitation." *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2012.
 - [18] GEAMBAŞU, Cristina Venera. "BPMN vs. UML Activity Diagram for Business Process Modeling." *AMIS 2012* (2012): 934.
 - [19] Nafie, Faisal Mohammed, and Samani A. Talab. "Comparative study between workflow tools Case study: Arabdox workflow and Bizagi express."
 - [20] Eriksson, Hans-Erik, and Magnus Penker. "Business modeling with UML." *Business Patterns at Work*, John Wiley & Sons, New York, USA (2000).
 - [21] Dijkman, Remco, Jorg Hofstetter, and Jana Koehler, eds. *Business Process Model and Notation*. Springer, 2011.
 - [22] <http://www.visual-paradigm.com/tutorials/from-business-process-to-use-cases.jsp> visited Jan-2016

-
- [23] Cruz, Estrela Ferreira, Ricardo J. Machado, and Maribel Yasmina Santos. "From business process models to use case models: A systematic approach." *Advances in Enterprise Engineering VIII*. Springer International Publishing, 2014. 167-181.
 - [24] OMG, Business process model and notation (BPMN), version 2.0," tech. rep., Object Management Group, 2011.
 - [25] Booch, Grady, James Rumbaugh, and Ivar Jacobson. "The unified modeling language user guide, 1999." MA: Addison-Wesley (2010).
 - [26] OMG, \Unified modeling language (OMG UML), version 2.5," Tech. Rep., Object Management Group, 2012.
 - [27] Pressman, Roger S. *Software engineering: a practitioner's approach*. Palgrave Macmillan, 2005.
 - [28] Bloch, Michael, Sven Blumberg, and Jürgen Laartz. "Delivering large-scale IT projects on time, on budget, and on value." *McKinsey Quarterly* (2012).
 - [29] Jalote, Pankaj. *A concise introduction to software engineering*. Springer Science & Business Media, 2008.
 - [30] Mili, Hafedh, et al. "Business process modeling languages: Sorting through the alphabet soup." *Manuscript*. November (2003).
 - [31] Cruz, Estrela Ferreira, Ricardo J. Machado, and Maribel Yasmina Santos. "From business process models to use case models: A systematic approach." *Advances in Enterprise Engineering VIII*. Springer International Publishing, 2014. 167-181.
 - [32] Rhazali, Y., Y. Hadi, and A. Mouloudi. "Transformation method CIM to PIM: from business processes models defined in BPMN to use case and class models defined in UML." *Transformation* 13609 (2014): 9999213.
 - [33] Demirörs, Onur, Çiğdem Gencel, and Ayça Tarhan. "Utilizing business process models for requirements elicitation." *Euromicro Conference, 2003. Proceedings. 29th. IEEE*, 2003.
 - [34] Coskuncay, Ahmet, et al. "Bridging the Gap between Business Process Modeling and Software Requirements Analysis: a Case Study." *MCIS*. 2010.

-
- [35] Van Lamsweerde, Axel. "Goal-oriented requirements engineering: A guided tour." *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*. IEEE, 2001.
- [36] Pohl, Klaus. *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated, 2010.
- [37] Rolland, Colette. "Capturing system intentionality with maps." *Conceptual modelling in Information Systems engineering*. Springer Berlin Heidelberg, 2007. 141-158.
- [38] Vernadat, François B. *Enterprise modelling and integration*. Springer US, 2003.
- [39] Loucopoulos, Pericles, and Vassilios Karakostas. *System requirements engineering*. McGraw-Hill, Inc., 1995.
- [40] Bubenko J.A., jr., A.Persson and J.Stirna. *EKD User Guide, (2001)*.
http://people.dsv.su.se/~js/ekd_user_guide.html Visited Jan 2016
- [41] Yu, Eric. "Modelling strategic relationships for process reengineering." *Social Modeling for Requirements Engineering* 11 (2011): 2011.
- [42] Moody, Daniel L., Patrick Heymans, and Raimundas Matulevičius. "Visual syntax does matter: improving the cognitive effectiveness of the i* visual notation." *Requirements Engineering* 15.2 (2010): 141-175.
- [43] Franch, Xavier. "Fostering the Adoption of i* by Practitioners: Some Challenges and Research Directions." *Intentional Perspectives on Information Systems Engineering*. Springer Berlin Heidelberg, 2010. 177-193.
- [44] España, Sergio, Arturo González, and Óscar Pastor. "Communication Analysis: a requirements engineering method for information systems." *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 2009.
- [45] Espana, Sergio, et al. "Systematic derivation of state machines from communication-oriented business process models." *Research Challenges in*

-
- Information Science (RCIS), 2011 Fifth International Conference on.* IEEE, 2011.
- [46] González, Arturo, et al. "Systematic derivation of class diagrams from communication-oriented business process models." *Enterprise, Business-Process and Information Systems Modeling*. Springer Berlin Heidelberg, 2011. 246-260.
- [47] Grover, Varun, and Manoj K. Malhotra. "Business process reengineering: A tutorial on the concept, evolution, method, technology and application." *Journal of operations management* 15.3 (1997): 193-213.
- [48] Reijers, Hajo A., and S. Liman Mansar. "Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics." *Omega* 33.4 (2005): 283-306.
- [49] Indulska, Marta, et al. "Business process modeling: Current issues and future challenges." *Advanced information systems engineering*. Springer Berlin Heidelberg, 2009.
- [50] Scheer, A.-W.: ARIS - Business Process Modeling. 3rd edn. Springer, Berlin, Germany , 2000
- [51] Aguilar-Saven, Ruth Sara. "Business process modelling: Review and framework." *International Journal of production economics* 90.2 (2004): 129-149.
- [52] Condori-Fernandez, Nelly, et al. , A systematic mapping study on empirical evaluation of software requirements specifications techniques. „Proceedings of the 2009 International Symposium on Empirical and Measurement, IEEE Computer society 2009.
- [53] Hove, Siw Elisabeth, and Bente Anda, Experiences from conducting semi structured interviews in empirical software engineering research,, Software Metrics, 2005 11th IEEE International Symposium IEEE 2005.
- [54] Anderson, Stuart, and Massimo Felici, Requirements engineering questionnaire , vol 1 2001:15

-
- [55] Dyba, Tore, Barbara A. Kitchenham, and Magne Jorgensen, Evidencebased software engineering for practitioners. *Software*, IEEE 22.1 2005 58-65.
- [56] Dieste, Oscar, Marta Lopez, and Felicidad Ramos. , Updating a Systematic Review about Selection of Software Requirements Elicitation Techniques. *WER* 2008
- [57] Gunda, Sai Ganesh. , Requirements engineering: elicitation techniques, 2008.
- [58] Maiden, N. A. M., and Gordon Rugg. , ACRE: selecting methods for requirements acquisition , *Software Engineering Journal* 11.3 (1996):183-192.
- [59] Yu, Lin Liu Eric., From Requirements to Architectural Design{ Using Goals and Scenarios , First International Workshop From Software Requirements to Architectures-STRAW. Vol.1. 2001.
- [60] Koch, Nora, and Martin Wirsing, Improving the quality of requirements with scenarios , *Proceedings of the Second World Congress on Software Quality*.2000.
- [61] OMG Unified Modeling Language: Superstructure Version 2.0 (online) <http://www.uml.org>. visited Feb 23, 2016.
- [62] Drake, José M., et al. "Approach and case study of requirement analysis where end users take an active role." *Proceedings of the 15th international conference on Software Engineering*. IEEE Computer Society Press, 1993.
- [63] Beichter, Friedrich W., Otthein Herzog, and Heiko Petzsch. "SLAN-4-A software specification and design language." *IEEE transactions on software engineering* 2 (1984): 155-162.
- [64] Rajagopal, Prasad, et al. "A new approach for software requirements elicitation." *Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Network*. IEEE, 2005.
- [65] Brooks, Fredrick P., Jr. "No Silver Bullet: Essence and Accidents of Software Engineering". *IEEE Computer*,10-19, April 1987.

-
- [66] Romney, Marshall. "Business process re-engineering." *The CPA Journal* 64.10 (1994): 30.
- [67] Larsen, Melissa A., and Michael D. Myers. "When success turns into failure: a package-driven business process re-engineering project in the financial services industry." *The Journal of Strategic Information Systems* 8.4 (1999): 395-417.
- [68] Hammer, Michael. "Reengineering work: don't automate, obliterate." *Harvard business review* 68.4 (1990): 104-112.
- [69] Ebenau, B. and Strauss, S., *Software Inspection Process*. McGraw-Hill, 1994.
- [70] ur Rehman, Tousif, Muhammad Naeem Ahmed Khan, and Naveed Riaz., *Analysis of Requirement Engineering Processes, Tools/Techniques and Methodologies., International Journal of Information Technology and Computer Science (IJITCS)* 5.3 2013:40
- [71] Bider, Ilia. "State-oriented business process modeling: principles, theory and practice." (2002).
- [72] Eriksson, H-E, Penker, M. *Business modeling with UML: business patterns at work*. John Wiley & Sons, Inc, 2000.
- [73] A. Cockburn, *Writing Effective Use Cases*. Addison Wesley, 2001.
- [74] D. Kulak and E. Guiney, *Use Cases: Requirements in Context*, ACM Press, 2000.
- [75] P. Kruchten: *The Rational Unified Process – an Introduction*, Addison-Wesley, 2000.
- [76] de la Vara González, Jose Luis. *Business process-based requirements specification and object-oriented conceptual modelling of information systems*. Diss. 2011.
- [77] Zowghi, Didar, and Chad Coulin. "Requirements elicitation: A survey of techniques, approaches, and tools." *Engineering and managing software requirements*. Springer Berlin Heidelberg, 2005. 19-46.
- [78] <http://www.sparxsystems.com.au/> visited Dec- 2016.

- [79] Christel, Michael G., and Kyo C. Kang. *Issues in requirements elicitation*. No. CMU/SEI-92-TR-12. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1992.
- [80] Davis, Alan, et al. "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review." *14th IEEE International Requirements Engineering Conference (RE'06)*. IEEE, 2006.
- [81] Neill, Colin J., and Phillip A. Laplante. "Requirements engineering: the state of the practice." *IEEE software* 20.6 (2003): 40.
- [82] Berry, Daniel M., and Erik Kamsties. "Ambiguity in requirements specification." *Perspectives on software requirements*. Springer US, 2004. 7-44.
- [83] Kitchenham, Barbara Ann, et al. "A framework for evaluating a software bidding model." *Information and Software Technology* 47.11 (2005): 747-760.

i. Appendix: Example how BMSpec generates Use case model.

This section includes two examples: request for proposal process and cab booking process, to show how BMSpec generates use case model:

- ***Request for proposal process***

The business process model shown in Figure 31, represents the Request for Proposal BPMN Process model. Generated use case diagram consist of the following items:

- Two performers in Request for Proposal process: Proposal writer and proposal approver. By **Rule2** the obtained use case diagram will have two actors with the corresponding name.
- Six activities will be transformed to six use cases by **Rule1**.
- Two data association will be transformed to include association using **Rule6**, **Rule7**, **Rule8** and **Rule9**.
- Three sequence flow between activity and decision gateway will be transformed to extend association, using **Rule5**.
- Two sequence flow between activities will be transformed to Precede association using **Rule4**

The obtained Request for proposal use case diagram is shown in Figure 32.

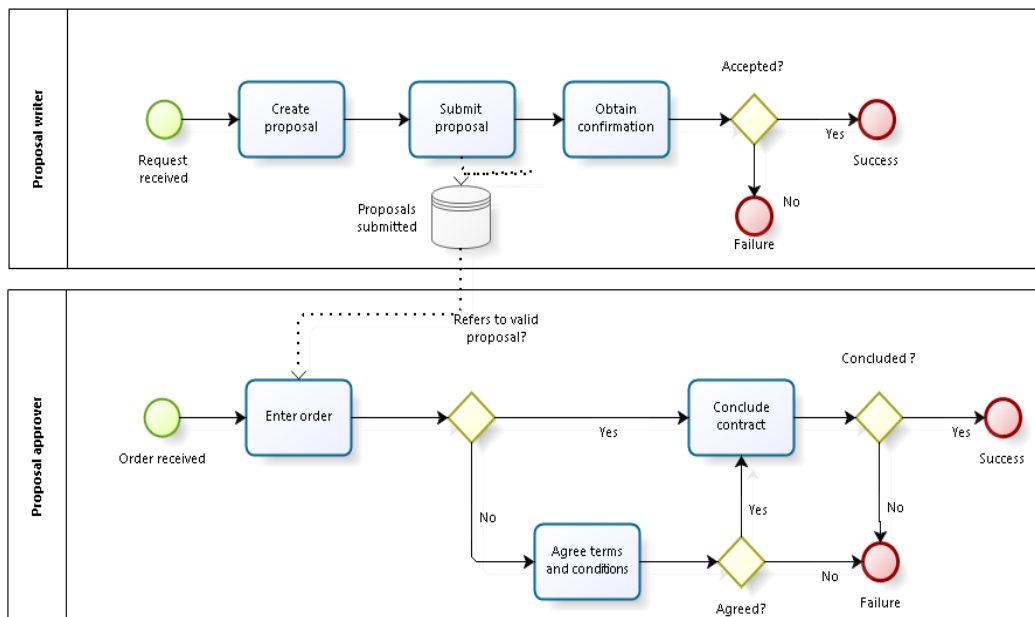


Figure 31: Request for Proposal process model.

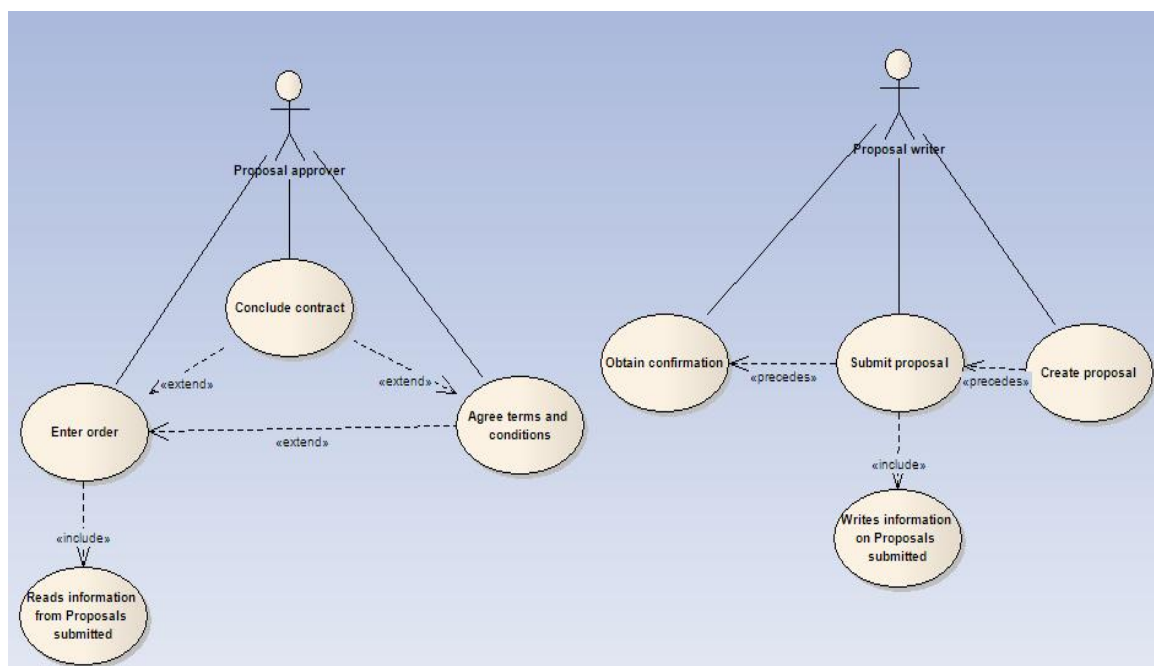


Figure 32: Request for proposal use case diagram.

Generated use case description consist of the following items distributed in six use case description:

- Two data association will be transformed to two sentences in scenario using **Rule11, Rule12, Rule13** and **Rule14**.
- Three sequence flow between activity and decision gateway will be transformed to three sentence in Pre-condition, using **Rule19**.
- Two sequence flow between activities will be two to sentences in Pre-condition using **Rule18**.
- Two sequence flow between start event and activity will be transformed to two trigger using **Rule27**.

The generated use case description are listed bellows:

Use Case Name	Create proposal
Actors	Proposal writer
Trigger	The Event Request received occurred
Pre-Condition	
Scenario	

Use Case Name	Submit proposal
Actors	Proposal writer
Trigger	
Pre-Condition	The Create proposal has been completed.
Scenario	Writes information on Proposals submitted

Use Case Name	Obtain confirmation
Actors	Proposal writer
Trigger	
Pre-Condition	The Submit proposal has been completed.
Scenario	

Use Case Name	Enter order
Actors	Proposal approver
Trigger	The Event Order received occurred
Pre-Condition	
Scenario	Reads information from Proposals submitted.

Use Case Name	Conclude contract
Actors	Proposal approver
Trigger	
Pre-Condition	The Refers to valid proposal? is Yes The Agreed? is Yes
Scenario	

Use Case Name	Agree terms and conditions
Actors	Proposal approver
Trigger	
Pre-Condition	The Refers to valid proposal? is No
Scenario	

- ***Cab booking process***

The business process model shown in Figure 33, represents the Cab booking BPMN Process model. Generated use case diagram consist of the following items:

- Three performers in Cab booking process: Customer, Travel Agent, and cab driver. By **Rule2** the obtained use case diagram will have three actors with the corresponding name.
- Eight activities will be transformed to eight use cases by **Rule1**.
- Three sequence flow between activity and decision gateway will be transformed to extend association, using **Rule5**.
- One sequence flow between activities will be transformed to Precede association using **Rule4**.
- One sequence flow between activity and service activity will be transformed to Invoke association using **Rule10**.

The obtained Cab booking use case diagram is shown in Figure 37.

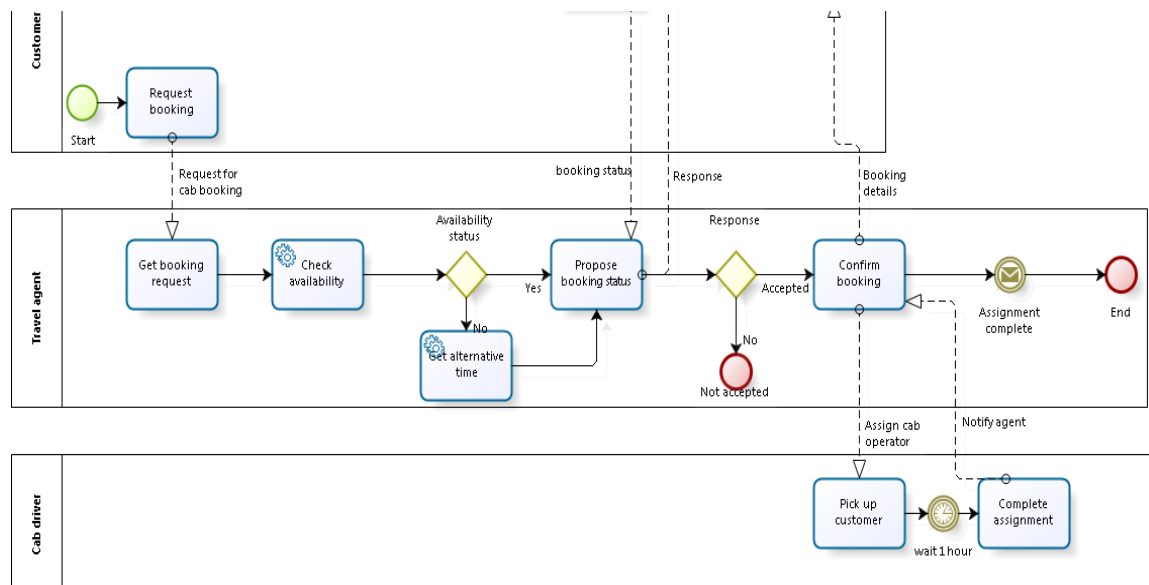


Figure 33: Cab booking BPMN Process model

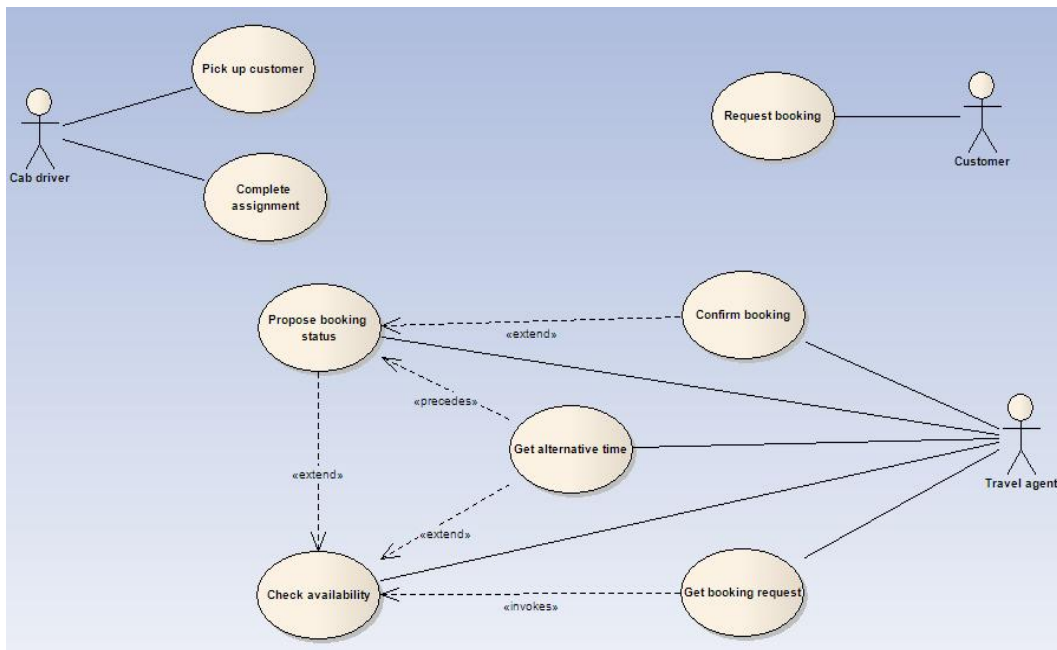


Figure 34: Cab booking use case diagram

Generated use case description consist of the following items distributed in eight use case description:

- Two message flow will be transformed to two sentences in scenario using **Rule16, Rule17**.
- Three sequence flow between activity and decision gateway will be transformed to three sentence in Pre-condition, using **Rule19**.
- Two sequence flow between activities will be two to sentences in Pre-condition using **Rule18**.
- One sequence flow between start event and activity will be transformed to two trigger using **Rule27**.

The generated use case description are listed bellows:

Use Case Name	Request booking
Actors	Customer
Trigger	The Event Start occurred
Pre-Condition	
Scenario	Send Request for cab booking To Travel agent

Use Case Name	Check availability
Actors	System
Trigger	
Pre-Condition	The Get booking request has been completed.
Scenario	

Use Case Name	Propose booking status
Actors	Customer
Trigger	
Pre-Condition	The Availability status is Yes The Get alternative time has been completed.
Scenario	Receive Response from Customer Send booking status To Customer

Use Case Name	Confirm booking
Actors	Cab driver
Trigger	
Post-Condition	The Message Assignment complete is send.
Pre-Condition	The Response is Accepted
Scenario	Receive Notify agent from Cab driver Send Assign cab operator To Cab driver

Use Case Name	Pick up customer
Actors	Travel agent
Trigger	
Post-Condition	The time-date wait 1 hour event is created.
Pre-Condition	
Scenario	Receive Assign cab operator from Travel agent

Use Case Name	Complete assignment
Actors	Cab driver
Trigger	The time-date wait 1 hour is reached.
Pre-Condition	
Scenario	Send Notify agent To Travel agent

ii. Appendix: Example of comparison between manual work and BMSpec results.

Extra feature calculator

According to manual work in extra feature calculator process the result use case diagram consists of two use cases with one actor, the same result we get from BMSpec as shown in table33, but BMSpec has extra features: Include association

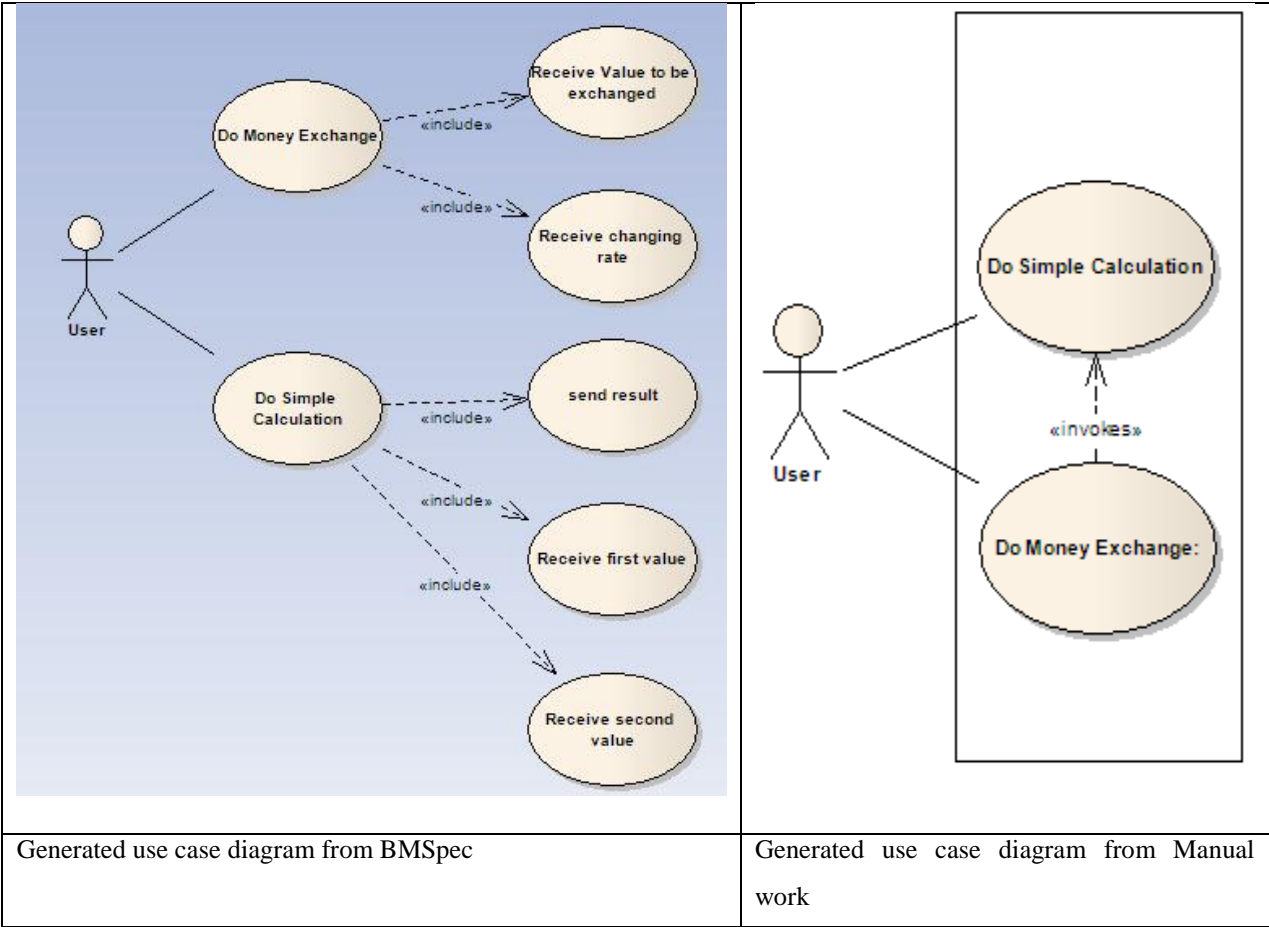


Figure : comparison between use case diagram generated from manual work and BMSpec for Calculator case.

In manual work they deal sometimes with the trigger as a step in scenario, they duplicate this information in trigger and in scenario. While in BMSpec we deal with it as only trigger, we don't duplicate this information twice, Another difference is in dealing with pre-condition, in manual work some of scenario

sentences appear also in the precondition, while in BMSpec it appear only in scenario. So we can notice different in number of sentences in scenario and precondition between BMSpec and manual work, but this difference is just in how deal with the same information and where to place it, generated use case description from manual work represented in figure35 and figure 36 , while generated use case description from BMSpec represented in table38 and table39. There is not any information loss. The correctness percentage is 100%.

1) Do Simple Calculation:	
1. Description:	the user enters first value and second value then press calculate, after that the system calculate the result and display it in result text box.
2. Actor:	user.
3. Pre-conditions:	first value and second value have been entered.
4. Sequence:	
a.	The user enters first value to be operated later.
b.	The user enters second value to be operated later.
c.	the user choose one of the operation :addition, subtraction, multiplication or division .
d.	the calculator operates depend on the selection of user.
e.	the calculator put the result in result Text box.
5. Trigger:	when the user click on the operation button +, -, *, =., also can be triggered from other use case such as do money exchange use case

Figure 35: Do Simple Calculation use case description from manual work

Use Case Name	Do Simple Calculation
Actors	User
Trigger	The Event user click on the operation button +, -, *, =. Occurred. The Do simple calculation Arrive from Do Money Exchange.
Post-Condition	Put the result in result Text box, The process ends
Pre-Condition	
Scenario	Receive second value. Receive first value. Send result.

Table 38: Simple Calculation use case description from BMSpe

2) Do Money Exchange:

1. **Description:** the user enters NIS value or Dollar value to be operate later, then enters changing rate, after that the user click calculate button. Then the calculator operates depend on the selection of user: if he enters Nis value it calculate Dollar value, if he enters Dollar value it calculate Nis value and put the result in result text box.
2. **Actor:**user
3. **Pre-conditions:** NIS value or Dollar value and changing rate have been entered.
4. **Sequence:**
 - a. the user enters NIS value or Dollar value to be operate later.
 - b. the user enters changing rate.
 - c. that the user click calculate button.
 - d. the calculator operates depend on the selection of user.
5. **Trigger:**when the user click on NIS to Dollar Button or Dollar to NIS button.

Figure 36: Do Money Exchange use case description from manual work

Use Case Name	Do Money Exchange
Actors	User
Trigger	The event click calculate button occurs.
Post-Condition	The Do simple calculation is send.
Pre-Condition	
Scenario	Receive Value to be exchanged. Receive changing rate.

Table 39: Do Money Exchange use case description from BMSpec

Registration on X-Road

According to manual work in X-road registration process, the result use case diagram consists of four use cases with two actors as shown in figure37, the same result we get from BMSpec, but with extra features: Include and extend association as shown in figure38.

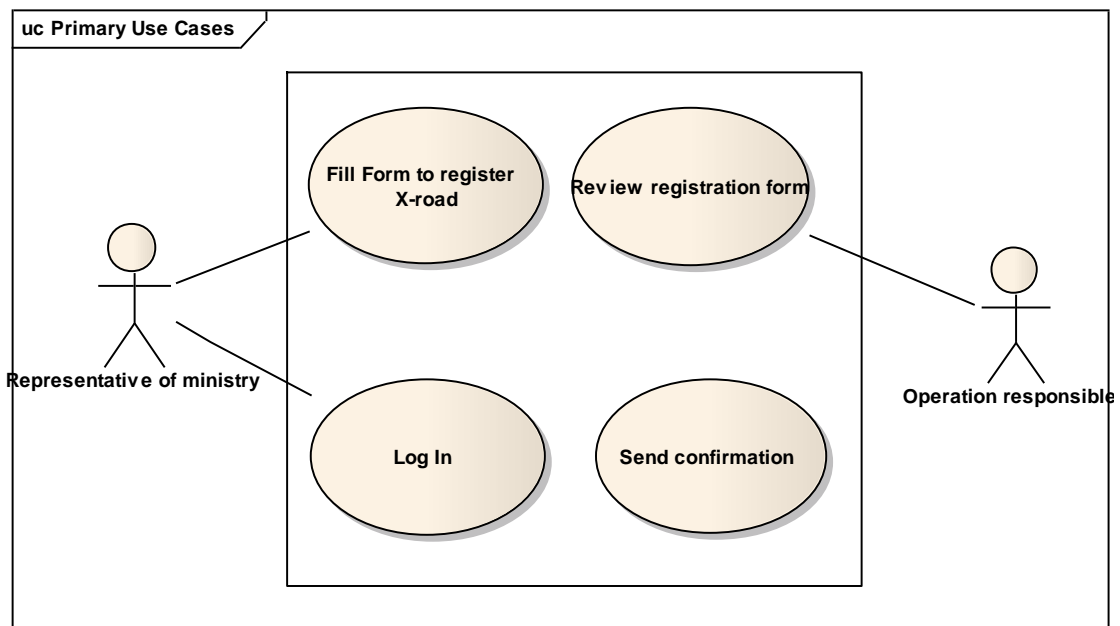


Figure 37: Use case diagram of X-road registration from manual work

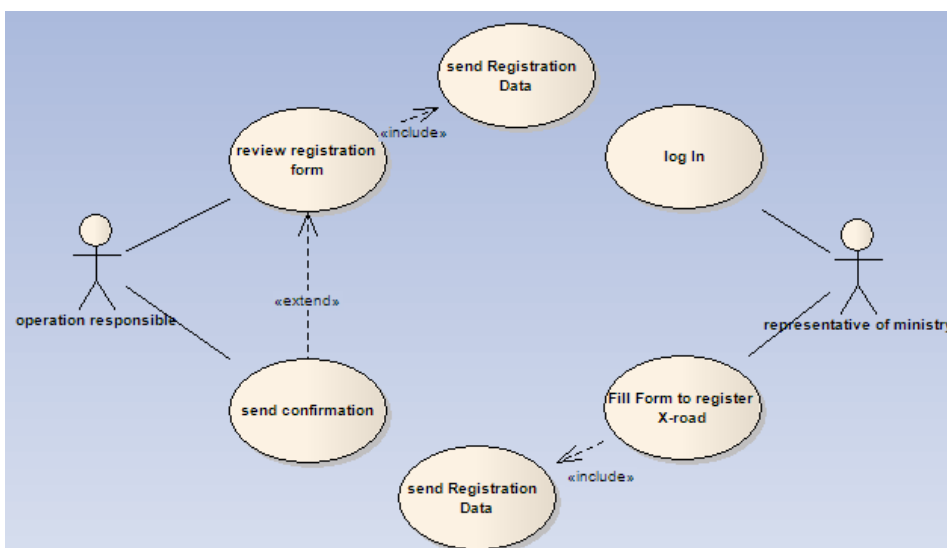


Figure 38: Use case diagram of X-road registration from BMSpec

Generated use case description from manual work represented in figure39 and figure40, while generated use case description from BMSpec represented in table40 and table41.

Use case	Fill Form to register X-road
actor	Representative of ministry
precondition	
Sequence	The user fill form of registration. The user Click submit button. The system save Registration Data. Send new form notification to the operation responsible.
trigger	The user click on register submit button

Figure 39: Fill Form to register X-road use case description from manual work.

Use Case Name	Fill Form to register X-road
Actors	representative of ministry
Trigger	The Event click register button occurred
Pre-Condition	
Scenario	Send Registration Data. Send new form notification to the operation responsible.

Table 40: Fill Form to register X-road use case description from BMSpec.

Use case	Send Confirmation
actor	Representative of ministry
precondition	The Acceptance is Yes
Sequence	The system Send confirmation Notification To representative of ministry
trigger	

Figure 40: send confirmation use case description from manual work

Use Case Name	send confirmation
Actors	operation responsible
Trigger	
Pre-Condition	The Acceptance is Yes
Scenario	Send confirmation Notification To representative of ministry

Table 41: send confirmation use case description from BMSpec.

Consume X-Road service

According to manual work in consume X-road service process, the result use case diagram consists of seven use cases with three actors as shown in figure41, the same result we get from BMSpec, but with extra features: Include, extend and precede association as shown in figure42.

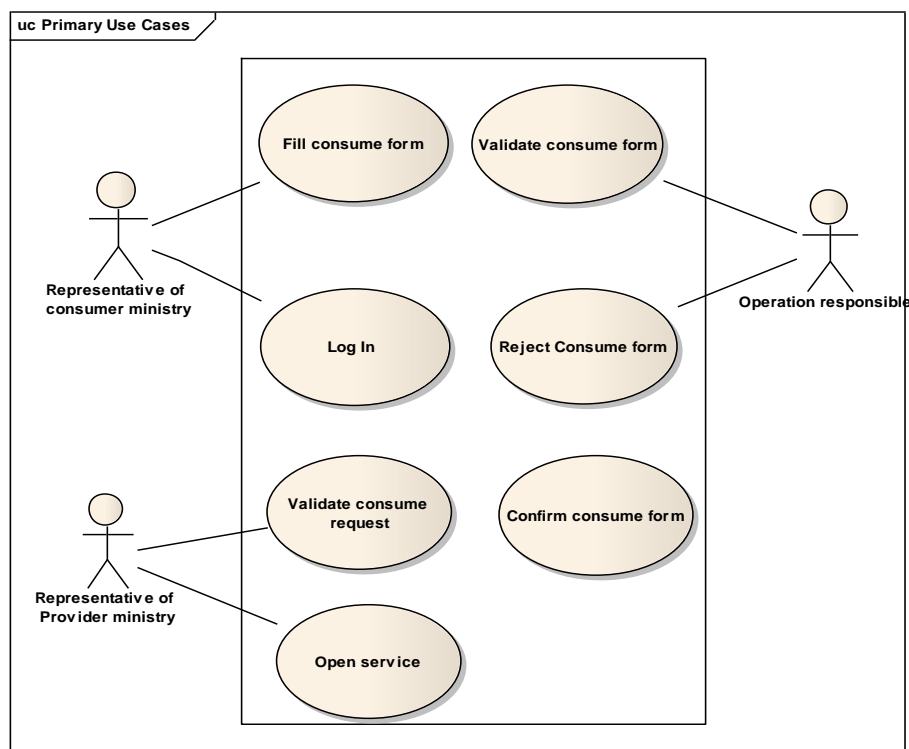


Figure 41: consume X-road service use case diagram from manual work

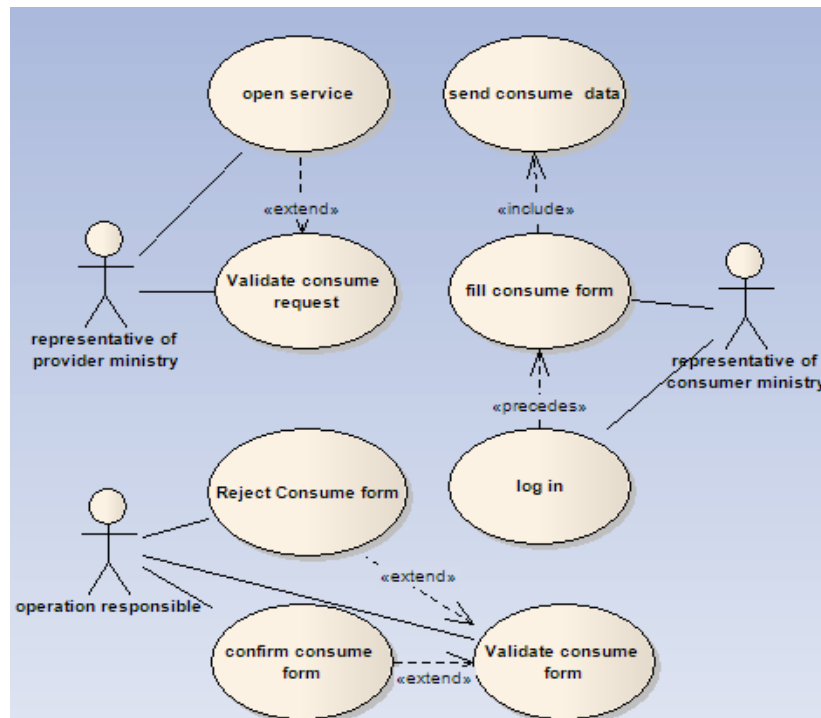


Figure 42: consume X-road service use case diagram from BMSpec

Generated use case description from manual work represented in figure43 and figure44, while generated use case description from BMSpec represented in table42 and table43.

Use case	Validate Consume Form
actor	Operation Responsible
precondition	
Sequence	Receive consume data. Receive new form notification.
trigger	Receive new form notification.

Figure 43: Validate consume form use case description from manual work

Use Case Name	Validate consume form
Actors	operation responsible
Trigger	The new form notification received
Pre-Condition	
Scenario	Receive consume data

Table 42: Validate consume form use case description from BMSpec.

Use case	Confirm Consume Form
actor	Operation Responsible
precondition	accept the consume form
Sequence	Send consume request To representative of provider ministry.
trigger	

Figure 44: Confirm consume form use case description from manual work

Use Case Name	confirm consume form
Actors	operation responsible
Trigger	
Pre-Condition	The Acceptance is Yes
Scenario	Send consume request To representative of provider ministry.

Table 43: Confirm consume form use case description from BMSpec.

Provide X-road service

According to manual work in provide X-road service process, the result use case diagram consists of five use cases with two actors as shown in figure45, the same result we get from BMSpec, but with extra features: Precede, Include and extend association as shown in figure46.

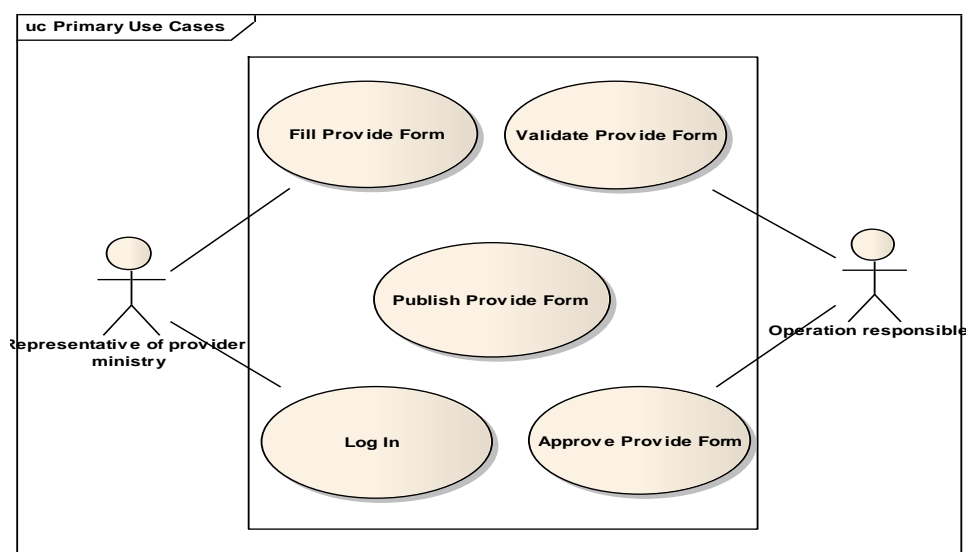


Figure 45: provide X-road service use case diagram from manual wor

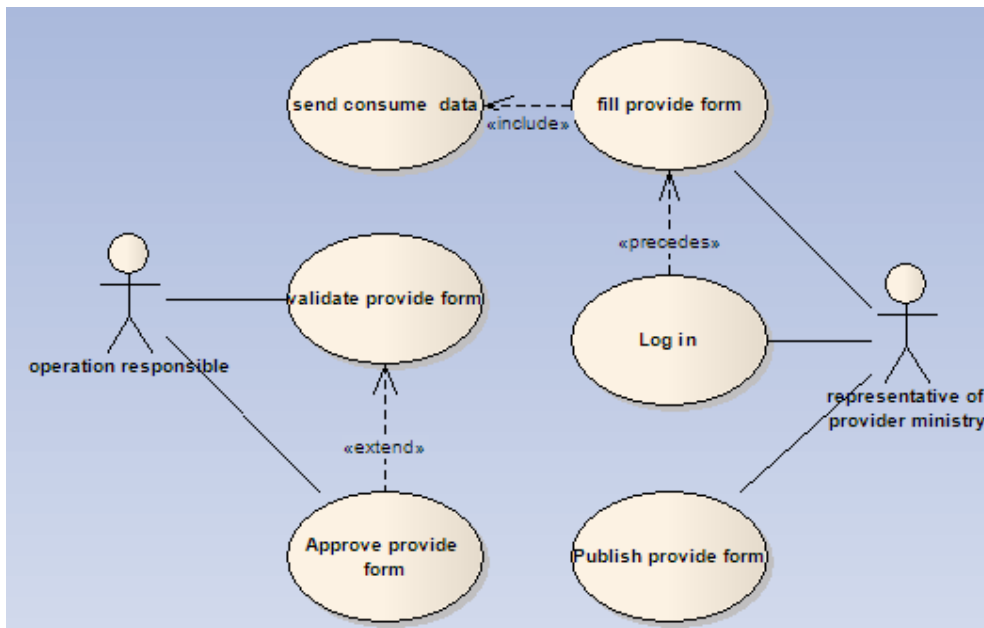


Figure 46: provide X-road service use case diagram from BMSpec.

Generated use case description from manual work represented in figure47 and figure48, while generated use case description from BMSpec represented in table44 and table45.

Use case	Fill Provide Form
actor	Representative of Provider Ministry
precondition	The Log in has been completed.
Sequence	Fill provide form. Send consume data. Send new form notification to operation responsible
trigger	

Figure 47: Fill provide form use case description from manual work.

Use Case Name	fill provide form
Actors	representative of provider ministry
Trigger	
Pre-Condition	The Log in has been completed.
Scenario	Send consume data. Send new form notification to operation responsible

Table 44: Fill provide form use case description from BMSpec

Use case	Publish provide form
actor	Representative of Provider Ministry
precondition	
Sequence	Receive Approve service notification from operation responsible. Publish the service.
trigger	

Figure 48: Publish provide form use case description from manual work.

Use Case Name	Publish provide form
Actors	representative of provider ministry
Trigger	
Post-Condition	The service published is created. The process ends.
Pre-Condition	
Scenario	Receive Approve service notification from operation responsible

Table 45: Publish provide form use case description from BMSpec.

Online Bookshop System(make order process)

According to manual work in make order process, the result use case diagram consists of five use cases with one actors as shown in figure49, the same result we get from BMSpec, but with extra features: Precede and Include association as shown in figure50.

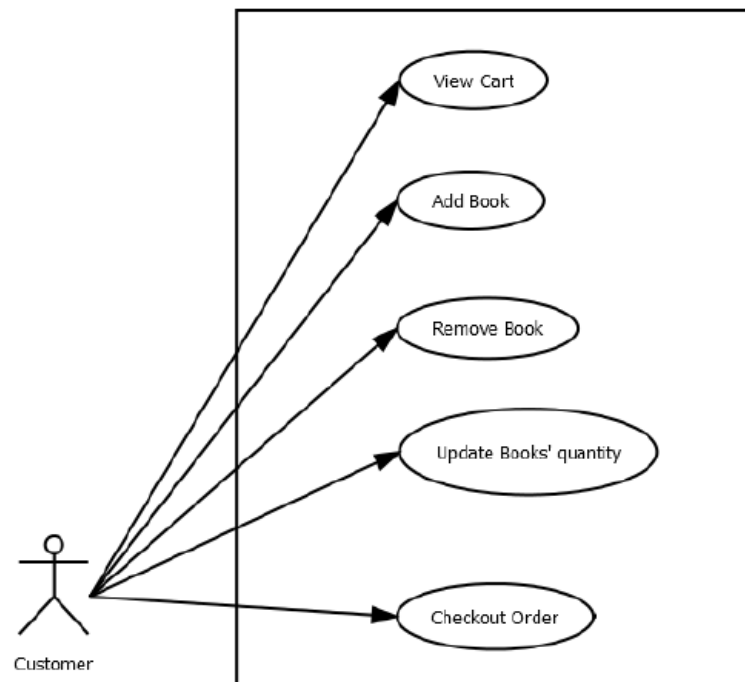


Figure 49: make order use case diagram from manual work

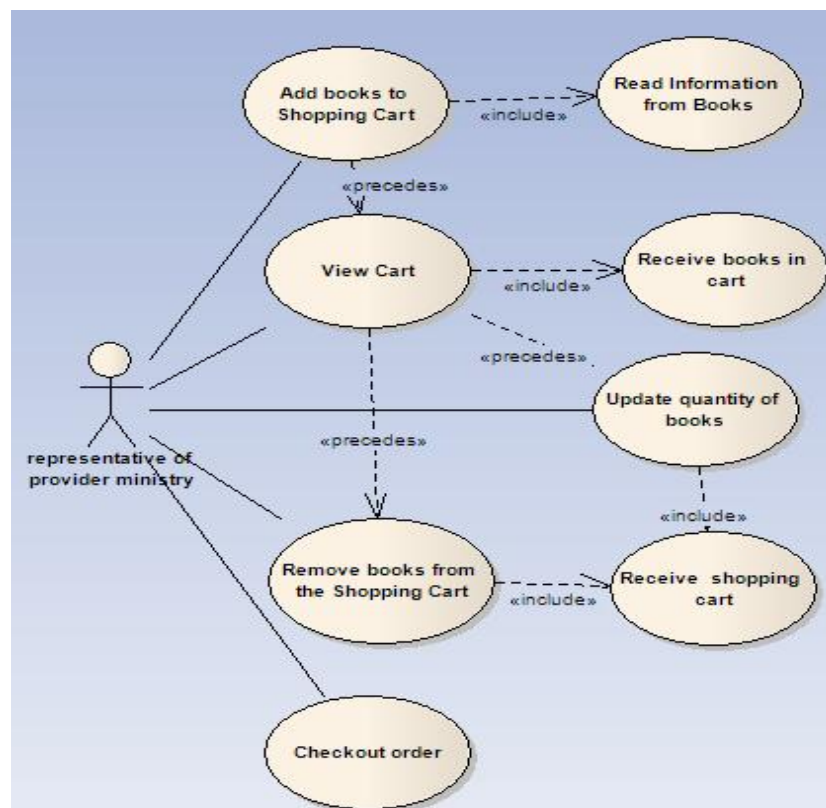


Figure 50: Figure 49: make order use case diagram from BMSpec

iii. Appendix: Gold standard cases

➤ Car Hire case study

It is a real case in which a team of four expert master degree students develop a software system for car Hire Company in web engineering course. The car hire system defined a web portal for the car hiring agencies, and the people looking for hiring cars, this system represents main marketing and facilitating system for car hiring agencies since it is a central point where many agencies will be offering cars for hiring. We use hire car process to evaluate BMSpec.

The team uses traditional requirement elicitation technique to understand system requirement. The team who works on this project is:

Mohammad ZeinEddin -- Project Manager email mohammad@zeineddin.name

Salam Turkman -- System analyst email salma.turk@gmail.com

Bashar Hammouri -- System Architect email eng.bash85@gmail.com

Abed Othman -- System Analyst email abedothman83@gmail.com

The output system requirement specification document contains a detailed description about the car hire system, it has a full details about what the system shall do and how it is expected to do it, The output is a UML use case model and a detailed use case description. The final work of the team has been checked and validated by the supervisor of the course.

➤ Online Bookshop System

It is a real case in which a team of four expert master degree students develop online bookshop system in web engineering course. Online bookshop system represents a bridge between the publishers or book suppliers and customers. It allows publishers or book suppliers to setup online shops, and it allows customers to browse and search through the shop and purchase them online without having to visit the shop physically. It makes the purchasing quicker, easier, and more convenient. We use make order process to evaluate BMSpec.

The team uses traditional requirement elicitation technique to understand system requirement. The team who works on this project is:

Ahmed Al-Jadaa	-- Project Manager	email a3.2015@outlook.com
Ahmed Alia	-- System analyst	email abualia4@yahoo.com
Abdeljawwad Daas	-- System Analyst	email abdeljawwad1@gmail.com

The output system requirement specification document contains a detailed description about the online bookshop system, it has a full details about what the system shall do and how it is expected to do it, and the output is a UML use case model and a detailed use case description. The final work of the team has been checked and validated by the supervisor of the course.

➤ **Extra feature calculator**

It is a real case in which a team of four expert master degree students develop Extra feature calculator in software engineering course. This Extra feature calculator includes a normal set of functions: addition, subtraction, multiplication, division, currency calculation (converting from dollar to NIS and vice versa). Also provides net salary and income tax calculation.

The team uses traditional requirement elicitation technique to understand system requirement. The team who works on this project is:

Salam Turkman	Project Manager	email salma.turk@gmail.com
Abdeljawwad Daas	System Analyst	email abdeljawwad1@gmail.com
Areen Salman --	System Analyst	email eng.bash85@gmail.com
Amani Abu Gharbiah	System Analyst	email abedothman83@gmail.com

The output system requirement specification document contains a detailed description about the Extra feature calculator, it has a full details about what the calculator shall do and how it is expected to do it, and the output is a UML use case model and a detailed use case description. The final work of the team has been checked and validated by the supervisor of the course.

➤ **Nobel Prize example**

It is a real case in which a paper work [31] used manual transformation from business process model for Nobel prize to a use case model, in their manual transformation they used a set of rules which we adopted in BMSpec.

We use the same business process model for Nobel prize used in [31], and apply BMSpec then compare the generate use case model with the manual transformation results.

➤ **X-road services in ministry of telecom and IT**

In ministry of Telecom and Information Technology we choose three real processes, these processes represent the daily work of registration on X-Road, Consume X-Road service and provide X-Road service. A team consists of two experts from the ministry use traditional requirement elicitation technique to understand system requirement.

The output system requirement specification document contains a detailed description about the X-road services, it has a full details about what the system shall do and how it is expected to do it, and the output is a UML use case model and a detailed use case description.

iv. Appendix: Business process modelling notation (BPMN)

Business process models consist of different types of graphical objects [24]:

- Flow objects, it's the main graphical objects for business process modelling, which includes three main objects: event, activity and gateway as represented in figure 51.



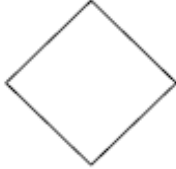
Flow Object Name	Notation
Event	
Activity	
Gateway	

Figure 51: Flow objects [24].

- Connecting objects, which allow other graphical objects to be connected, they are sequence flows (normal and default), message flows and associations as represented in figure52.


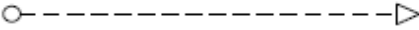

Connecting Objects Name	Notation
Sequence Flow	
Message Flow	
Association	

Figure 52: Connecting objects [24]

- Swimlanes, which allow participants of a business process to be represented, they are pools and lanes as represented in figure53.


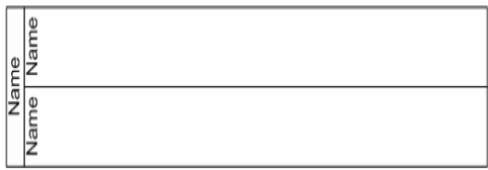
Swimlanes Objects Name	Notation
Pool	
Lane	

Figure 53: Swimlanes [24]

- Artifacts, which provide additional information about a business process model; they are data objects, groups and annotations as represented in figure54.



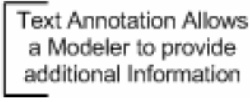
Artifacts Objects Name	Notation
Data Object	
Group	
Annotation	

Figure 54: Artifacts [24]

v. Appendix: source code for generating use case diagram

```
Dim xmlDoc As New XmlDataDocument()
Dim xmlNode As XmlNodeList
Dim xmlPerformernode As XmlNodeList
Dim xmlAssociationlistNode As XmlNodeList
Dim xmldataSeqnode As XmlNode
Dim DataObjects As XmlNodeList
Dim i As Integer
Dim artifactID As String
Dim performerID As String
Dim performername As String
Dim source As String
Dim usecase1 As String
Dim usecase2 As String
Dim usecaseID1 As String
Dim usecaseID2 As String
Dim eventNode As XmlNode

Dim gateWayType As String

Dim usecaseID As String
Dim usecasename As String
Dim xmlDataStorelistNode As XmlNodeList
Dim xmldatainputnode As XmlNode
Dim xmlDataAssosListnode As XmlNodeList
Dim xmlDataStoreRefnode As XmlNodeList
Dim inputnode As XmlNode
Dim fs As New FileStream("/data/nobelpz.xpd1", FileMode.Open, FileAccess.Read)
xmlDoc.Load(fs)
xmlNode = xmlDoc.GetElementsByTagName("Activity")
xmlPerformernode = xmlDoc.GetElementsByTagName("Participant")
xmlDataAssosListnode = xmlDoc.GetElementsByTagName("DataAssociation")
xmlDataStoreRefnode = xmlDoc.GetElementsByTagName("DataStoreReference")
xmlDataStorelistNode = xmlDoc.GetElementsByTagName("DataStore")
xmlAssociationlistNode = xmlDoc.GetElementsByTagName("Association")
DataObjects = xmlDoc.GetElementsByTagName("DataObject")
Dim writer As New XmlTextWriter("/data/ucase25.xml", System.Text.Encoding.GetEncoding(1252))

writer.WriteStartDocument(False)
writer.Formatting = Formatting.Indented
writer.Indentation = 2

writer.WriteDocType("XMI", "System", "/data/UML_EA.dtd", "")
writer.WriteStartElement("XMI")
writer.WriteAttributeString("xmi.version", "1.1")
writer.WriteAttributeString("xmlns:UML", "omg.org/UML1.3")
writer.WriteStartElement("XMI.header")
writer.WriteStartElement("XMI.documentation")
writer.WriteStartElement("XMI.exporter")
writer.WriteString("Enterprise Architect")
writer.WriteEndElement()
writer.WriteStartElement("XMI.exporterVersion")
writer.WriteString("2.5")
writer.WriteEndElement()
writer.WriteEndElement()
writer.WriteEndElement()
```

```
writer.WriteStartElement("XMI.content")
writer.WriteStartElement("UML:Model")
' writer.WriteAttributeString("xmi.id", "MX_EAID_9DB7A453_B385_4278_ADC2_EF4B733C5ECC")
writer.WriteStartElement("UML:Namespace.ownedElement")
writer.WriteStartElement("UML:Package")
writer.WriteAttributeString("xmi.id", "EAPK_Package_Main")
writer.WriteAttributeString("name", "Use Case Model")
writer.WriteStartElement("UML:Namespace.ownedElement")

For k = 0 To xmlnode.Count - 1
    usecaseID = xmlnode(k).Attributes(0).InnerText.Trim()
    usecasename = xmlnode(k).Attributes(1).InnerText.Trim()

    If usecasename <> "" Then
        'use case
        writer.WriteStartElement("UML:UseCase")
        writer.WriteAttributeString("xmi.id", "EAID_" & usecaseID)
        writer.WriteAttributeString("name", usecasename)
        writer.WriteStartElement("UML:ModelElement.taggedValue")
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_stype")
        writer.WriteAttributeString("value", "UseCase")
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()
        'end 'use case
    End If

    If Not IsNothing(xmlnode(k).Item("InputSets")) Then
        artifactID = ""
        xmldatainputnode = xmlnode(k).Item("InputSets")
        If xmldatainputnode.HasChildNodes() Then

            xmldataSeqnode = xmldatainputnode.Item("InputSet")
            If (xmldataSeqnode.HasChildNodes()) Then
                For Each node As XmlNode In xmldataSeqnode.ChildNodes
                    artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                    If artifactID <> "" Then
                        For j = 0 To xmlDataAssosListNode.Count - 1

                            If artifactID = xmlDataAssosListNode(j).Attributes("To").InnerText.Trim() Then

                                Dim dataRefid As String =
xmlDataAssosListNode(j).Attributes("From").InnerText.Trim()

                                For m = 0 To xmlDataStoreRefnode.Count - 1
                                    If dataRefid = xmlDataStoreRefnode(m).Attributes("Id").InnerText.Trim() Then

                                        Dim dataref As String =
xmlDataStoreRefnode(m).Attributes("DataStoreRef").InnerText.Trim()

                                        For s = 0 To xmlDataStoreListNode.Count - 1
                                            If dataref =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() Then

                                                'included use cas
                                                usecase2 = " Reads information from " &
xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim()
                                                usecaseID2 =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim()
```

[illegible]


```

        usecase2 = " Writes information on " &
xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim()
        usecaseID2 =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() & "MM"

        ' MsgBox(xmlNode(k).Attributes("Name").InnerText.Trim() & " " &
usecase2)

        writer.WriteStartElement("UML:UseCase")
        writer.WriteAttributeString("xmi.id", "EAID_" & usecaseID2)
        writer.WriteAttributeString("name", usecase2)
        writer.WriteStartElement("UML:ModelElement.taggedValue")
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_stype")
        writer.WriteAttributeString("value", "UseCase")
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()

    End If
Next
End If
Next
End If
Next
End If
End If
End If

'out data set use case
usecaseID = xmlNode(k).Attributes(0).InnerText.Trim()
For j = 0 To xmlAssociationListNode.Count - 1
    If usecaseID = xmlAssociationListNode(j).Attributes("Source").InnerText.Trim() Then
        artifactID = xmlAssociationListNode(j).Attributes("Target").InnerText.Trim()
        For m = 0 To DataObjects.Count - 1

            If artifactID = DataObjects(m).Attributes("Id").InnerText.Trim() Then

                ' datastore = " send " & DataObjects(m).Attributes("Name").InnerText.Trim()
                usecase2 = " send " & DataObjects(m).Attributes("Name").InnerText.Trim()
                usecaseID2 = DataObjects(m).Attributes("Id").InnerText.Trim() & "D" & j

                ' MsgBox(xmlNode(k).Attributes("Name").InnerText.Trim() & " " & usecase2)
                writer.WriteStartElement("UML:UseCase")
                writer.WriteAttributeString("xmi.id", "EAID_" & usecaseID2)
                writer.WriteAttributeString("name", usecase2)
                writer.WriteStartElement("UML:ModelElement.taggedValue")
                writer.WriteStartElement("UML:TaggedValue")
                writer.WriteAttributeString("tag", "ea_stype")
                writer.WriteAttributeString("value", "UseCase")
                writer.WriteEndElement()
                writer.WriteEndElement()
                writer.WriteEndElement()

                ' performerID = xmlPerfnode.InnerText

```

```

        End If
    Next
End If
Next

'first we must get input data set BPMN2.2
If Not IsNothing(xmlnode(k).Item("InputSets")) Then
    xmldatainputnode = xmlnode(k).Item("InputSets")
    If xmldatainputnode.HasChildNodes() Then

        xmldataSeqnode = xmldatainputnode.Item("InputSet")
        If (xmldataSeqnode.HasChildNodes()) Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes
                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListnode.Count - 1

                        If artifactID = xmlDataAssosListnode(j).Attributes("To").InnerText.Trim() Then

                            Dim dataRefid As String =
                                xmlDataAssosListnode(j).Attributes("From").InnerText.Trim()

                            For z = 0 To DataObjects.Count - 1

                                If dataRefid = DataObjects(z).Attributes("Id").InnerText.Trim() Then
                                    usecase2 = " Receive " &
                                        DataObjects(z).Attributes("Name").InnerText.Trim()
                                    usecaseID2 = DataObjects(z).Attributes("Id").InnerText.Trim() & "I" & j

                                    ' MsgBox(xmlnode(k).Attributes("Name").InnerText.Trim() & " " &
                                usecase2)

                                    writer.WriteStartElement("UML:UseCase")
                                    writer.WriteAttributeString("xmi.id", "EAID_" & usecaseID2)
                                    writer.WriteAttributeString("name", usecase2)
                                    writer.WriteStartElement("UML:ModelElement.taggedValue")
                                    writer.WriteStartElement("UML:TaggedValue")
                                    writer.WriteAttributeString("tag", "ea_stype")
                                    writer.WriteAttributeString("value", "UseCase")
                                    writer.WriteEndElement()
                                    writer.WriteEndElement()
                                    writer.WriteEndElement()

                                    ' datastore = " Receive " &
                                DataObjects(z).Attributes("Name").InnerText.Trim()
                            End If
                        Next
                    End If
                Next
            End If
        Next
    End If
End If
'end of get inputset BPMN2.2

Next
```

```
For g = 0 To xmlPerformernode.Count - 1
    performerID = xmlPerformernode(g).Attributes(0).InnerText.Trim()
    performername = xmlPerformernode(g).Attributes(1).InnerText.Trim()
    'actor
    writer.WriteStartElement("UML:Actor")
    writer.WriteAttributeString("xmi.id", "EAID_" & performerID)
    writer.WriteAttributeString("name", performername)
    writer.WriteStartElement("UML:ModelElement.taggedValue")
    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("value", "Actor")
    writer.WriteAttributeString("tag", "ea_stype")
    writer.WriteEndElement()
    writer.WriteEndElement()
    writer.WriteEndElement()
    'end actor
Next
Dim xmlchildnode As XmlNode

Dim xmlTransitionListNode As XmlNodeList

xmlTransitionListNode = xmldoc.GetElementsByTagName("Transition")
'start gateway
For k = 1 To xmlnode.Count - 1

    usecaseID = xmlnode(k).Attributes("Id").InnerText.Trim()
    For j = 0 To xmlTransitionListNode.Count - 1
        If usecaseID = xmlTransitionListNode(j).Attributes("To").InnerText.Trim() Then
            source = xmlTransitionListNode(j).Attributes("From").InnerText.Trim()
            For g As Integer = 0 To xmlnode.Count - 1
                If source = xmlnode(g).Attributes("Id").InnerText.Trim() Then
                    If Not IsNothing(xmlnode(g).Item("Event")) Then
                        'event
                    ElseIf Not IsNothing(xmlnode(g).Item("Route")) Then
                        Dim routeElement As XmlElement = xmlnode(g).Item("Route")
                        Dim OuttransitionCoun As Integer = 0
                        Dim gatewayID = xmlnode(g).Attributes("Id").InnerText.Trim()
                        For m = 0 To xmlTransitionListNode.Count - 1
                            If gatewayID = xmlTransitionListNode(m).Attributes("To").InnerText.Trim() Then

                                End If
                                If gatewayID = xmlTransitionListNode(m).Attributes("From").InnerText.Trim() Then
                                    OuttransitionCoun += 1
                                End If
                            Next
                            If (routeElement.HasAttribute("GatewayType")) Then
                                gatewayType = routeElement.GetAttribute("GatewayType")

                                If gatewayType = "Parallel" Then
                                ElseIf gatewayType = "Inclusive" Then
                                ElseIf gatewayType = "Complex" Then
                                Else
                                End If

                            Else

```

```

        If OuttransitionCoun > 1 Then
            'extend relation
            For m = 0 To xmlTransitionListnode.Count - 1

                If gatewayID = xmlTransitionListnode(m).Attributes("To").InnerText.Trim()

Then
                    Dim sourceActivity =
xmlTransitionListnode(m).Attributes("From").InnerText.Trim()

                    For c As Integer = 0 To xmlnode.Count - 1
                        If sourceActivity = xmlnode(c).Attributes("Id").InnerText.Trim()

Then

                            If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "") Then

                                usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
                                usecase2 = xmlnode(c).Attributes("Name").InnerText.Trim()
                                usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
                                usecaseID2 = xmlnode(c).Attributes("Id").InnerText.Trim()
                                ' asiciation
                                writer.WriteStartElement("UML:Association")
                                writer.WriteAttributeString("xmi.id", "EAID_Association" & j

                                writer.WriteStartElement("UML:ModelElement.taggedValue")

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "style")
                                writer.WriteAttributeString("value", "3")
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "ea_type")
                                writer.WriteAttributeString("value", "UseCase")
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "direction")
                                writer.WriteAttributeString("value", "Destination -&gt;

Source")

                                writer.WriteEndElement()

                            'type extend
                            writer.WriteStartElement("UML:TaggedValue")
                            writer.WriteAttributeString("tag", "subtype")
                            writer.WriteAttributeString("value", "Extends")
                            writer.WriteEndElement()

                            'type extend
                            writer.WriteStartElement("UML:TaggedValue")
                            writer.WriteAttributeString("tag", "stereotype")
                            writer.WriteAttributeString("value", "extend")
                            writer.WriteEndElement()

                            'type extend
                            writer.WriteStartElement("UML:TaggedValue")

```

```
        writer.WriteString("tag", "conditional")
        writer.WriteString("value", " &#xA;«extend»")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteString("tag", "ea_sourceName")
        writer.WriteString("value", usecase2)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteString("tag", "ea_targetName")
        writer.WriteString("value", usecase1)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteString("tag", "ea_sourceType")
        writer.WriteString("value", "UseCase")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteString("tag", "ea_targetType")
        writer.WriteString("value", "UseCase")
        writer.WriteEndElement()

        'extend
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteString("tag", "mb")
        writer.WriteString("value", " &#xA;«extend»")
        writer.WriteEndElement()

        writer.WriteEndElement()

        'connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteString("type", "EAID_" & usecaseID2)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteString("type", "EAID_" & usecaseID1)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()

        'end association

        End If
    End If
Next
End If
Next
Else

End If

Else
    'we need to check if it is service task then the relation is invoke else it is include.
```

```
eventNode = xmlnode(k).Item("Implementation")
If Not IsNothing(eventNode) Then

    If eventNode.HasChildNodes() Then

        Dim tasknode As XmlNode = eventNode.Item("Task")
        If tasknode.HasChildNodes() Then
            If Not IsNothing(tasknode.Item("TaskService")) Then
                'invoke relation
                If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "" And
xmlnode(g).Attributes("Name").InnerText.Trim() <> "") Then

                    usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
                    usecase2 = xmlnode(g).Attributes("Name").InnerText.Trim()
                    usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
                    usecaseID2 = xmlnode(g).Attributes("Id").InnerText.Trim()
                    '' asiciation
                    writer.WriteStartElement("UML:Association")
                    writer.WriteAttributeString("xmi.id", "EAID_Association" & i & "_" &
g & "INV")

                    writer.WriteStartElement("UML:ModelElement.taggedValue")

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "style")
                    writer.WriteAttributeString("value", "3")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_type")
                    writer.WriteAttributeString("value", "Dependency")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "direction")
                    writer.WriteAttributeString("value", "Source -> Destination")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "stereotype")
                    writer.WriteAttributeString("value", "invokes")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_sourceName")
                    writer.WriteAttributeString("value", usecase2)
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_targetName")
                    writer.WriteAttributeString("value", usecase1)
                    writer.WriteEndElement()


```

```
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_sourceType")
        writer.WriteAttributeString("value", "UseCase")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_targetType")
        writer.WriteAttributeString("value", "UseCase")
        writer.WriteEndElement()

        'extend
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "mb")
        writer.WriteAttributeString("value", " &#xA;«invokes»")
        writer.WriteEndElement()

        writer.WriteEndElement()

        'connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID2)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID1)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()
        'end association
    'end route
End If
End If
Else
    'precede relation

    If (xmlNode(k).Attributes("Name").InnerText.Trim() <> "" And
xmlNode(g).Attributes("Name").InnerText.Trim() <> "") Then

        If Not IsNothing(xmlNode(k).Item("Event")) Then
        Else
            usecase1 = xmlNode(k).Attributes("Name").InnerText.Trim()
            usecase2 = xmlNode(g).Attributes("Name").InnerText.Trim()
            usecaseID1 = xmlNode(k).Attributes("Id").InnerText.Trim()
            usecaseID2 = xmlNode(g).Attributes("Id").InnerText.Trim()
            ' asiciation
            writer.WriteStartElement("UML:Association")
            writer.WriteAttributeString("xmi.id", "EAID_Association" & i & "_" &

g & "N")

            writer.WriteStartElement("UML:ModelElement.taggedValue")

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "style")
            writer.WriteAttributeString("value", "3")
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "ea_type")
            writer.WriteAttributeString("value", "Dependency")
```

```
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "direction")
        writer.WriteAttributeString("value", "Source -&gt; Destination")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "stereotype")
        writer.WriteAttributeString("value", "precedes")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_sourceName")
        writer.WriteAttributeString("value", usecase2)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_targetName")
        writer.WriteAttributeString("value", usecase1)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_sourceType")
        writer.WriteAttributeString("value", "UseCase")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_targetType")
        writer.WriteAttributeString("value", "UseCase")
        writer.WriteEndElement()

        ''extend
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "mb")
        writer.WriteAttributeString("value", " &#xA;«precedes»")
        writer.WriteEndElement()

        writer.WriteEndElement()

        ''connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID2)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID1)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()
        '' end association
    'end route
End If

End If

End If
```



```
Else
    'precede relation

    If (xmlNode(k).Attributes("Name").InnerText.Trim() <> "" And
xmlNode(g).Attributes("Name").InnerText.Trim() <> "") Then

        If Not IsNothing(xmlNode(k).Item("Event")) Then
Else
    usecase1 = xmlNode(k).Attributes("Name").InnerText.Trim()
    usecase2 = xmlNode(g).Attributes("Name").InnerText.Trim()
    usecaseID1 = xmlNode(k).Attributes("Id").InnerText.Trim()
    usecaseID2 = xmlNode(g).Attributes("Id").InnerText.Trim()
    ' asiciation
    writer.WriteStartElement("UML:Association")
    writer.WriteAttributeString("xmi.id", "EAID_Association" & i & "_" & g &
"N")

    writer.WriteStartElement("UML:ModelElement.taggedValue")

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "style")
    writer.WriteAttributeString("value", "3")
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "ea_type")
    writer.WriteAttributeString("value", "Dependency")
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "direction")
    writer.WriteAttributeString("value", "Source -> Destination")
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "stereotype")
    writer.WriteAttributeString("value", "precedes")
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "ea_sourceName")
    writer.WriteAttributeString("value", usecase2)
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "ea_targetName")
    writer.WriteAttributeString("value", usecase1)
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "ea_sourceType")
    writer.WriteAttributeString("value", "UseCase")
    writer.WriteEndElement()

    writer.WriteStartElement("UML:TaggedValue")
    writer.WriteAttributeString("tag", "ea_targetType")
    writer.WriteAttributeString("value", "UseCase")
    writer.WriteEndElement()
```

```
        'extend
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "mb")
        writer.WriteAttributeString("value", " &#xA;«precedes»")
        writer.WriteEndElement()

        writer.WriteEndElement()

        'connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID2)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID1)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()
        'end association
    'end route
End If

End If
End If
End If
End If
Next
End If
Next

Next

For k = 0 To xmlnode.Count - 1

    If Not IsNothing(xmlnode(k).Item("InputSets")) Then
        artifactID = ""
        xmldatainputnode = xmlnode(k).Item("InputSets")
        If xmldatainputnode.HasChildNodes() Then

            xmldataSeqnode = xmldatainputnode.Item("InputSet")
            If (xmldataSeqnode.HasChildNodes()) Then
                For Each node As XmlNode In xmldataSeqnode.ChildNodes

                    artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                    If artifactID <> "" Then
                        For j = 0 To xmlDataAssosListnode.Count - 1

                            If artifactID = xmlDataAssosListnode(j).Attributes("To").InnerText.Trim() Then

                                Dim dataRefid As String =
xmlDataAssosListnode(j).Attributes("From").InnerText.Trim()

                                For m = 0 To xmlDataStoreRefnode.Count - 1
                                    If dataRefid = xmlDataStoreRefnode(m).Attributes("Id").InnerText.Trim() Then

                                        Dim dataref As String =
xmlDataStoreRefnode(m).Attributes("DataStoreRef").InnerText.Trim()
```

```

        For s = 0 To xmlDataStoreListNode.Count - 1
            If dataref =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() Then
                If (xmlNode(k).Attributes("Name").InnerText.Trim() <> "") Then
                    ' sequence = " reads information from " &
xmlDataStoreListNode(s).Attributes("name").InnerText.Trim()
                    usecase1 = xmlNode(k).Attributes("Name").InnerText.Trim()
                    usecase2 = " reads information from " &
xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim()
                    usecaseID1 = xmlNode(k).Attributes("Id").InnerText.Trim()
                    usecaseID2 =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim()
                    'include relation
                    ' asiciation
                    writer.WriteStartElement("UML:Association")
                    writer.WriteAttributeString("xmi.id", "EAID_Association" & i
& "_" & s & "SS")

                    writer.WriteStartElement("UML:ModelElement.taggedValue")

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "style")
                    writer.WriteAttributeString("value", "3")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_type")
                    writer.WriteAttributeString("value", "usecase")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "direction")
                    writer.WriteAttributeString("value", "Source ->
Destination")

                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "subtype")
                    writer.WriteAttributeString("value", "includes")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "stereotype")
                    writer.WriteAttributeString("value", "include")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_sourcename")
                    writer.WriteAttributeString("value", usecase1)
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_targetname")
                    writer.WriteAttributeString("value", usecase2)
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")
                    writer.WriteAttributeString("tag", "ea_sourcetype")
                    writer.WriteAttributeString("value", "usecase")
                    writer.WriteEndElement()

                    writer.WriteStartElement("UML:TaggedValue")

```

```
        writer.WriteString("tag", "ea_targettype")
        writer.WriteString("value", "usecase")
        writer.WriteEndElement()

        'extend
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteString("tag", "mb")
        writer.WriteString("value", " &#xa;<include>")
        writer.WriteEndElement()
        writer.WriteEndElement()

        'connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteString("type", "EAID_" & usecaseID1)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteString("type", "EAID_" & usecaseID2)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()

        'add performer to the included task

    End If
End If
Next
End If
Next
End If
Next
End If
Next
End If
End If

' sequence output data store BPMN2.2
If Not IsNothing(xmlnode(k).Item("OutputSets")) Then
    artifactID = ""
    xmldatainputnode = xmlnode(k).Item("OutputSets")
    If xmldatainputnode.HasChildNodes() Then
        xmldataSeqnode = xmldatainputnode.Item("OutputSet")

        If xmldataSeqnode.HasChildNodes() Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes
                ' inputnode = node.Item("Output")
                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListnode.Count - 1

                        If artifactID = xmlDataAssosListnode(j).Attributes("From").InnerText.Trim() Then

                            Dim dataRefid As String =
                                xmlDataAssosListnode(j).Attributes("To").InnerText.Trim()

                            For m = 0 To xmlDataStoreRefnode.Count - 1
                                If dataRefid = xmlDataStoreRefnode(m).Attributes("Id").InnerText.Trim() Then
```

```
Dim dataref As String =
xmlDataStoreRefNode(m).Attributes("DataStoreRef").InnerText.Trim()
For s = 0 To xmlDataStoreListNode.Count - 1
    If dataref =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() Then

        'included use cas

        If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "") Then
            usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
            usecase2 = " Writes information on " &

xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim()

            usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
            usecaseID2 =

xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() & "MM"

            'include relation

            ' asiciation
            writer.WriteStartElement("UML:Association")
            writer.WriteAttributeString("xmi.id", "EAID_Association" & i

& "_" & s & "MM")

            writer.WriteStartElement("UML:ModelElement.taggedValue")

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "style")
            writer.WriteAttributeString("value", "3")
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "ea_type")
            writer.WriteAttributeString("value", "usecase")
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "direction")
            writer.WriteAttributeString("value", "Source -&gt;

Destination")

            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "subtype")
            writer.WriteAttributeString("value", "includes")
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "stereotype")
            writer.WriteAttributeString("value", "include")
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "ea_sourcename")
            writer.WriteAttributeString("value", usecase1)
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "ea_targetname")
            writer.WriteAttributeString("value", usecase2)
            writer.WriteEndElement()

            writer.WriteStartElement("UML:TaggedValue")
            writer.WriteAttributeString("tag", "ea_sourcetype")
```



```
writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "style")
writer.WriteAttributeString("value", "3")
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "ea_type")
writer.WriteAttributeString("value", "usecase")
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "direction")
writer.WriteAttributeString("value", "Source -> Destination")
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "subtype")
writer.WriteAttributeString("value", "includes")
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "stereotype")
writer.WriteAttributeString("value", "include")
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "ea_sourcename")
writer.WriteAttributeString("value", usecase1)
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "ea_targetname")
writer.WriteAttributeString("value", usecase2)
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "ea_sourcetype")
writer.WriteAttributeString("value", "usecase")
writer.WriteEndElement()

writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "ea_targettype")
writer.WriteAttributeString("value", "usecase")
writer.WriteEndElement()

'extend
writer.WriteStartElement("UML:TaggedValue")
writer.WriteAttributeString("tag", "mb")
writer.WriteAttributeString("value", " &#xa;«include»")
writer.WriteEndElement()

writer.WriteEndElement()

'connection
writer.WriteStartElement("UML:Association.connection")
writer.WriteStartElement("UML:AssociationEnd")
writer.WriteAttributeString("type", "EAID_" & usecaseID1)
writer.WriteEndElement()
writer.WriteStartElement("UML:AssociationEnd")
writer.WriteAttributeString("type", "EAID_" & usecaseID2)
```

```
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()
    End If

End If

Next

End If

Next

'input data set

If Not IsNothing(xmlnode(k).Item("InputSets")) Then
    xmldatainputnode = xmlnode(k).Item("InputSets")
    If xmldatainputnode.HasChildNodes() Then

        xmldataSeqnode = xmldatainputnode.Item("InputSet")
        If (xmldataSeqnode.HasChildNodes()) Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes

                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListnode.Count - 1
                        If artifactID = xmlDataAssosListnode(j).Attributes("To").InnerText.Trim() Then
                            Dim dataRefid As String =
                                xmlDataAssosListnode(j).Attributes("From").InnerText.Trim()
                            For z = 0 To DataObjects.Count - 1
                                If dataRefid = DataObjects(z).Attributes("Id").InnerText.Trim() Then
                                    usecase2 = " Receive " &
                                        DataObjects(z).Attributes("Name").InnerText.Trim()
                                    usecaseID2 = DataObjects(z).Attributes("Id").InnerText.Trim() & "I" & j

                                    If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "") Then
                                        usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
                                        usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
                                        usecase2 = " Receive " &
                                            DataObjects(z).Attributes("Name").InnerText.Trim()
                                        usecaseID2 = DataObjects(z).Attributes("Id").InnerText.Trim() & "I"
                                    & j

                                    writer.WriteStartElement("UML:Association")
                                    writer.WriteAttributeString("xml.id", "EAID_Association" & i & "_" &
                                        z & "I" & j)

                                    writer.WriteStartElement("UML:ModelElement.taggedValue")
                                    writer.WriteStartElement("UML:TaggedValue")
                                    writer.WriteAttributeString("tag", "style")
                                    writer.WriteAttributeString("value", "3")
                                    writer.WriteEndElement()

                                    writer.WriteStartElement("UML:TaggedValue")
                                    writer.WriteAttributeString("tag", "ea_type")
                                    writer.WriteAttributeString("value", "usecase")
                                    writer.WriteEndElement()

                                    writer.WriteStartElement("UML:TaggedValue")
                                    writer.WriteAttributeString("tag", "direction")
                                    writer.WriteAttributeString("value", "Source -> Destination")
                                    writer.WriteEndElement()

                                    writer.WriteStartElement("UML:TaggedValue")
                                    writer.WriteAttributeString("tag", "subtype")
                                    writer.WriteAttributeString("value", "includes")
                                End If
                            End For
                        End If
                    End For
                End If
            End For
        End If
    End If
End If
```



```
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "stereotype")
        writer.WriteAttributeString("value", "include")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_sourcename")
        writer.WriteAttributeString("value", usecase1)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_targetname")
        writer.WriteAttributeString("value", usecase2)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_sourcetype")
        writer.WriteAttributeString("value", "usecase")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_targettype")
        writer.WriteAttributeString("value", "usecase")
        writer.WriteEndElement()

        ''extend
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "mb")
        writer.WriteAttributeString("value", " &#xa;«include»")
        writer.WriteEndElement()

        writer.WriteEndElement()

        ''connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID1)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID2)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()

    End If

End If

Next

End If

Next

End If

Next

End If

End If

Next
```

```
For i = 0 To xmlnode.Count - 1
    'we need to get list of performers for each activities
    If Not IsNothing(xmlnode(i).Item("Performers")) Then

        xmlchildnode = xmlnode(i).Item("Performers")
        If xmlchildnode.HasChildNodes() Then
            For c = 0 To xmlchildnode.ChildNodes.Count - 1

                performerID = xmlchildnode.ChildNodes(c).InnerText.Trim()
                If performerID <> "" Then
                    usecaseID = xmlnode(i).Attributes(0).InnerText.Trim()
                    usecasename = xmlnode(i).Attributes(1).InnerText.Trim()
                    If usecasename <> "" Then

                        For j = 0 To xmlPerformernode.Count - 1
                            If performerID = xmlPerformernode(j).Attributes(0).InnerText.Trim() Then
                                performername = xmlPerformernode(j).Attributes(1).InnerText.Trim()
                                ' MsgBox(usecasename & " do" & performername)
                                ' asiciation
                                writer.WriteStartElement("UML:Association")
                                writer.WriteAttributeString("xmi.id", "EAID_Association" & i & "_" & c)
                                writer.WriteStartElement("UML:ModelElement.taggedValue")

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "style")
                                writer.WriteAttributeString("value", "3")
                                writer.WriteEndElement()
                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "ea_type")
                                writer.WriteAttributeString("value", "UseCase")
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "direction")
                                writer.WriteAttributeString("value", "Source -&gt; Destination")
                                writer.WriteEndElement()
                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "linemode")
                                writer.WriteAttributeString("value", "3")
                                writer.WriteEndElement()
                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "linecolor")
                                writer.WriteAttributeString("value", "-1")
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "linewidth")
                                writer.WriteAttributeString("value", "0")
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "seqno")
                                writer.WriteAttributeString("value", "0")
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:TaggedValue")
                                writer.WriteAttributeString("tag", "headStyle")
                                writer.WriteAttributeString("value", "0")
                                writer.WriteEndElement()
                            End If
                        Next j
                    End If
                End If
            Next c
        End If
    End If
Next i
```

```
        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "lineStyle")
        writer.WriteAttributeString("value", "0")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_localid")
        writer.WriteAttributeString("value", "4")
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_sourceName")
        writer.WriteAttributeString("value", performername)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:TaggedValue")
        writer.WriteAttributeString("tag", "ea_targetName")
        writer.WriteAttributeString("value", usecasename)
        writer.WriteEndElement()
        writer.WriteEndElement()
        'connection
        writer.WriteStartElement("UML:Association.connection")
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & performerID)
        writer.WriteEndElement()
        writer.WriteStartElement("UML:AssociationEnd")
        writer.WriteAttributeString("type", "EAID_" & usecaseID)
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.WriteEndElement()
        'end association
    End If
Next
End If
End If
End If
Next

End If
Next
writer.WriteEndElement()
writer.WriteEndElement()
writer.WriteEndElement()
writer.WriteEndElement()
'diagram
writer.WriteStartElement("UML:Diagram")
writer.WriteAttributeString("name", "Use Case Model")
writer.WriteAttributeString("xmi.id", "EAID_415A6C5F_2617_4efd_A572_2CD71A7332A2")
writer.WriteAttributeString("diagramType", "UseCaseDiagram")
writer.WriteAttributeString("owner", "EAPK_Package_Main")
writer.WriteAttributeString("toolName", "Enterprise Architect 2.5")

writer.WriteStartElement("UML:Diagram.element")
For i = 0 To xmlnode.Count - 1
```

```
'all performer
If Not IsNothing(xmlnode(i).Item("Performers")) Then
    xmlchildnode = xmlnode(i).Item("Performers")
    If xmlchildnode.HasChildNodes() Then
        For c = 0 To xmlchildnode.ChildNodes.Count - 1

            performerID = xmlchildnode.ChildNodes(c).InnerText.Trim()

            If performerID <> "" Then
                usecaseID = xmlnode(i).Attributes(0).InnerText.Trim()
                usecasename = xmlnode(i).Attributes(1).InnerText.Trim()
                If usecasename <> "" Then
                    For j = 0 To xmlPerformernode.Count - 1
                        If performerID = xmlPerformernode(j).Attributes(0).InnerText.Trim() Then
                            performername = xmlPerformernode(j).Attributes(1).InnerText.Trim()
                            writer.StartElement("UML:DiagramElement")
                            writer.AttributeString("geometry",
"Left=292;Top=141;Right=397;Bottom=211;")
                            writer.AttributeString("subject", "EAID_" & usecaseID)
                            writer.EndElement()

                            writer.StartElement("UML:DiagramElement")
                            writer.AttributeString("geometry",
"Left=175;Top=113;Right=220;Bottom=203;")
                            writer.AttributeString("subject", "EAID_" & performerID)
                            writer.EndElement()

                            writer.StartElement("UML:DiagramElement")
                            writer.AttributeString("geometry",
"SX=0;SY=0;EX=0;EY=0;EDGE=2;$LLB=;LLT=;LMT=;LMB=;LRT=;LRB=;IRHS=;ILHS=;Path=;")
                            writer.AttributeString("subject", "EAID_Association" & i)
                            writer.EndElement()

                        End If
                    Next
                End If
            End If
        Next

    End If
End If
Next

'draw extends

For k = 0 To xmlnode.Count - 1
    usecaseID = xmlnode(k).Attributes("Id").InnerText.Trim()
    For j = 0 To xmlTransitionListNode.Count - 1
        If usecaseID = xmlTransitionListNode(j).Attributes("To").InnerText.Trim() Then
            source = xmlTransitionListNode(j).Attributes("From").InnerText.Trim()
            For g As Integer = 0 To xmlnode.Count - 1
                If source = xmlnode(g).Attributes("Id").InnerText.Trim() Then
                    If Not IsNothing(xmlnode(g).Item("Event")) Then
                        ElseIf Not IsNothing(xmlnode(g).Item("Route")) Then
                            Dim routeElement As XmlElement = xmlnode(g).Item("Route")
                            Dim OuttransitionCoun As Integer = 0
                            Dim gatewayID = xmlnode(g).Attributes("Id").InnerText.Trim()
                            For m = 0 To xmlTransitionListNode.Count - 1
```

```
        If gatewayID = xmlTransitionListNode(m).Attributes("To").InnerText.Trim() Then

        End If

        If gatewayID = xmlTransitionListNode(m).Attributes("From").InnerText.Trim() Then
            OuttransitionCoun += 1
        End If
    Next
    If (routeElement.HasAttribute("GatewayType")) Then
        gateWayType = routeElement.GetAttribute("GatewayType")

        If gateWayType = "Parallel" Then
        ElseIf gateWayType = "Inclusive" Then
        ElseIf gateWayType = "Complex" Then
        Else
        End If

    Else

        If OuttransitionCoun > 1 Then

            For m = 0 To xmlTransitionListNode.Count - 1
                If gatewayID = xmlTransitionListNode(m).Attributes("To").InnerText.Trim()

                Then

                    Dim sourceActivity =
xmlTransitionListNode(m).Attributes("From").InnerText.Trim()

                    For c As Integer = 0 To xmlNode.Count - 1
                        If sourceActivity = xmlNode(c).Attributes("Id").InnerText.Trim()

                        Then

                            If (xmlNode(k).Attributes("Name").InnerText.Trim() <> "") Then
                                ' MsgBox(xmlNode(k).Attributes("Name").InnerText.Trim() & "
extends " & xmlNode(c).Attributes("Name").InnerText.Trim())

                                usecase1 = xmlNode(k).Attributes("Name").InnerText.Trim()
                                usecase2 = xmlNode(c).Attributes("Name").InnerText.Trim()
                                usecaseID1 = xmlNode(k).Attributes("Id").InnerText.Trim()
                                usecaseID2 = xmlNode(c).Attributes("Id").InnerText.Trim()
                                '' asiciation
                                writer.WriteStartElement("UML:DiagramElement")
                                writer.WriteAttributeString("geometry",

"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0;LRT=;LRB=;IRHS=;ILHS=;Path=;")

                                writer.WriteAttributeString("subject", "EAID_Association" &

j & "_" & c & "EE" & m & j)

                                writer.WriteEndElement()

                            End If
                        End If
                    Next
                End If
            Next
        Else

        End If

    End If

    End If

    'if service draw invoke else precede
    eventNode = xmlNode(k).Item("Implementation")
    If Not IsNothing(eventNode) Then
```

```
        If eventNode.HasChildNodes() Then

            Dim tasknode As XmlNode = eventNode.Item("Task")
            If tasknode.HasChildNodes() Then
                If Not IsNothing(tasknode.Item("TaskService")) Then
                    'invoke relation
                    '    MsgBox(xmlnode(k).Attributes("Name").InnerText.Trim() & "is
invoked")

                    If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "" And
xmlnode(g).Attributes("Name").InnerText.Trim() <> "") Then
                        usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
                        usecase2 = xmlnode(g).Attributes("Name").InnerText.Trim()
                        usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
                        usecaseID2 = xmlnode(g).Attributes("Id").InnerText.Trim()
                        '' asiciation

                        writer.WriteStartElement("UML:DiagramElement")
                        writer.WriteAttributeString("geometry",
"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0;LRT=;LRB=;IRHS=;ILHS=;Path=;")

                        writer.WriteAttributeString("subject", "EAID_Association" & i & "_"
& g & "INV")

                        writer.WriteEndElement()
                    End If
                Else
                    'precede
                    If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "" And
xmlnode(g).Attributes("Name").InnerText.Trim() <> "") Then

                        usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
                        usecase2 = xmlnode(g).Attributes("Name").InnerText.Trim()
                        usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
                        usecaseID2 = xmlnode(g).Attributes("Id").InnerText.Trim()
                        '' asiciation

                        writer.WriteStartElement("UML:DiagramElement")
                        writer.WriteAttributeString("geometry",
"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0;LRT=;LRB=;IRHS=;ILHS=;Path=;")

                        writer.WriteAttributeString("subject", "EAID_Association" & j & "_"
& g & "S")

                        writer.WriteEndElement()
                    End If
                End If
            End If
        Else
            If (xmlnode(k).Attributes("Name").InnerText.Trim() <> "" And
xmlnode(g).Attributes("Name").InnerText.Trim() <> "") Then

                usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()
                usecase2 = xmlnode(g).Attributes("Name").InnerText.Trim()
                usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()
                usecaseID2 = xmlnode(g).Attributes("Id").InnerText.Trim()
                '' asiciation

                writer.WriteStartElement("UML:DiagramElement")
                writer.WriteAttributeString("geometry",
"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0;LRT=;LRB=;IRHS=;ILHS=;Path=;")
```

[illegible]

```

writer.WriteStartElement("UML:DiagramElement")
writer.WriteAttributeString("geometry",

"Left=292;Top=141;Right=397;Bottom=211;")

writer.WriteAttributeString("subject", "EAID_" & usecaseID2)
writer.WriteEndElement()

writer.WriteStartElement("UML:DiagramElement")
writer.WriteAttributeString("geometry",

"SX=0;SY=0;EX=0;EY=0;EDGE=2;$LLB=;LLT=;LMT=;LMB=;LRT=;LRB=;IRHS=;ILHS=;Path=;")
writer.WriteAttributeString("subject", "EAID_Association" &

s)

writer.WriteEndElement()
End If

End If

Next

End If

Next

End If

Next

End If

End If

End If

' sequence output data store BPMN2.2
If Not IsNothing(xmlnode(k).Item("OutputSets")) Then
    artifactID = ""
    xmldatainputnode = xmlnode(k).Item("OutputSets")
    If xmldatainputnode.HasChildNodes() Then
        xmldataSeqnode = xmldatainputnode.Item("OutputSet")

        If xmldataSeqnode.HasChildNodes() Then
            ' we need for loop

            For Each node As XmlNode In xmldataSeqnode.ChildNodes
                ' inputnode = node.Item("Output")
                artifactID = node.Attributes("ArtifactId").InnerText.Trim()

                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListNode.Count - 1

                        If artifactID = xmlDataAssosListNode(j).Attributes("From").InnerText.Trim() Then

                            Dim dataRefid As String =
xmlDataAssosListNode(j).Attributes("To").InnerText.Trim()

                            For m = 0 To xmlDataStoreRefnode.Count - 1
                                If dataRefid = xmlDataStoreRefnode(m).Attributes("Id").InnerText.Trim() Then

                                    Dim dataref As String =
xmlDataStoreRefnode(m).Attributes("DataStoreRef").InnerText.Trim()
                                    For s = 0 To xmlDataStoreListNode.Count - 1
                                        If dataref =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() Then
```



```

                                usecase2 = " Writes information On " &
xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim()
                                usecaseID2 =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() & "MM"
                                ' here we can continue
                                If (usecase2 <> "" And
xmlnode(k).Attributes("Name").InnerText.Trim() <> "") Then

                                usecase1 = xmlnode(k).Attributes("Name").InnerText.Trim()

                                usecaseID1 = xmlnode(k).Attributes("Id").InnerText.Trim()

                                '' asiciation
                                writer.WriteStartElement("UML:DiagramElement")
                                writer.WriteAttributeString("geometry",
"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0:LRT=;LRB=;IRHS=;ILHS=;Path=;")

                                writer.WriteAttributeString("subject", "EAID_Association" &
"EAID_Association" & i & "_" & s & "MM")

                                writer.WriteEndElement()
                                writer.WriteStartElement("UML:DiagramElement")
                                writer.WriteAttributeString("geometry",
"Left=292;Top=141;Right=397;Bottom=211;")

                                writer.WriteAttributeString("subject", "EAID_" & usecaseID2)
                                writer.WriteEndElement()

                                writer.WriteStartElement("UML:DiagramElement")
                                writer.WriteAttributeString("geometry",
"SX=0;SY=0;EX=0;EY=0;EDGE=2;$LLB=;LLT=;LMT=;LMB=;LRT=;LRB=;IRHS=;ILHS=;Path=;")
                                writer.WriteAttributeString("subject", "EAID_Association" &
s)

                                writer.WriteEndElement()
                                End If

                                End If
                                Next
                                End If
                                Next

                                End If
                                Next
                                End If
                                Next
                                End If

                                End If
                                ,
                                End If

                                usecaseID = xmlnode(k).Attributes(0).InnerText.Trim()
                                For j = 0 To xmlAssociationListNode.Count - 1
                                    If usecaseID = xmlAssociationListNode(j).Attributes("Source").InnerText.Trim() Then
                                        artifactID = xmlAssociationListNode(j).Attributes("Target").InnerText.Trim()
                                        For m = 0 To DataObjects.Count - 1

                                            If artifactID = DataObjects(m).Attributes("Id").InnerText.Trim() Then
                                                usecase2 = " send " & DataObjects(m).Attributes("Name").InnerText.Trim()
                                                usecaseID2 = DataObjects(m).Attributes("Id").InnerText.Trim() & "D" & j
                                                writer.WriteStartElement("UML:DiagramElement")

```

```
        writer.WriteAttributeString("geometry",
"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0;LRT=;LRB=;IRHS=;ILHS=;Path=;")
        writer.WriteAttributeString("subject", "EAID_Association" & "EAID_Association" & i & "_" & j
& "D")

        writer.WriteEndElement()

        writer.WriteStartElement("UML:DiagramElement")
        writer.WriteAttributeString("geometry", "Left=292;Top=141;Right=397;Bottom=211;")
        writer.WriteAttributeString("subject", "EAID_" & usecaseID2)
        writer.WriteEndElement()

        writer.WriteStartElement("UML:DiagramElement")
        writer.WriteAttributeString("geometry",
"SX=0;SY=0;EX=0;EY=0;EDGE=2;$LLB=;LLT=;LMT=;LMB=;LRT=;LRB=;IRHS=;ILHS=;Path=;")
        writer.WriteAttributeString("subject", "EAID_Association" & m)
        writer.WriteEndElement()
    End If
Next
End If
Next

'first we must get input data set BPMN2.2
If Not IsNothing(xmlnode(k).Item("InputSets")) Then
    xmldatainputnode = xmlnode(k).Item("InputSets")
    If xmldatainputnode.HasChildNodes() Then

        xmldataSeqnode = xmldatainputnode.Item("InputSet")
        If (xmldataSeqnode.HasChildNodes()) Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes

                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListnode.Count - 1

                        If artifactID = xmlDataAssosListnode(j).Attributes("To").InnerText.Trim() Then

                            Dim dataRefid As String =
xmlDataAssosListnode(j).Attributes("From").InnerText.Trim()

                            For z = 0 To DataObjects.Count - 1

                                If dataRefid = DataObjects(z).Attributes("Id").InnerText.Trim() Then
                                    usecase2 = " Receive " &
DataObjects(z).Attributes("Name").InnerText.Trim()
                                    usecaseID2 = DataObjects(z).Attributes("Id").InnerText.Trim() & "I" & j

                                    writer.WriteStartElement("UML:DiagramElement")
                                    writer.WriteAttributeString("geometry",
"EDGE=4;$LLB=;LLT=;LMT=;LMB=CX=45:CY=14:OX=0:OY=0:HDN=0:BLD=0:ITA=0:UND=0:CLR=-
1:ALN=0:DIR=0:ROT=0;LRT=;LRB=;IRHS=;ILHS=;Path=;")
                                    writer.WriteAttributeString("subject", "EAID_Association" &
"EAID_Association" & i & "_" & z & "I" & j)
                                    writer.WriteEndElement()

                                    writer.WriteStartElement("UML:DiagramElement")
                                    writer.WriteAttributeString("geometry",
"Left=292;Top=141;Right=397;Bottom=211;")
                                    writer.WriteAttributeString("subject", "EAID_" & usecaseID2)
```

```
        writer.WriteEndElement()

        writer.WriteStartElement("UML:DiagramElement")
        writer.WriteAttributeString("geometry",
        "SX=0;SY=0;EX=0;EY=0;EDGE=2;$LLB=;LLT=;LMT=;LMB=;LRT=;LRB=;IRHS=;ILHS=;Path=;")
        writer.WriteAttributeString("subject", "EAID_Association" & z)
        writer.WriteEndElement()

        ' performerID = xmlPerfnode.InnerText
    End If
Next
End If
Next
End If
Next
End If
End If
'end of get inputset BPMN2.2
Next
```

vi. Appendix: source code for generating use case description

```
Dim xmlDoc As New XmlDocument()
Dim doc As New XmlDocument()

Dim xmlNode As XmlNodeList
Dim xmlChildNode As XmlNode
Dim inputNode As XmlNode
Dim dataObject As XmlNode
Dim xmlDataInputNode As XmlNode
Dim xmlPerfNode As XmlNode
Dim xmlDataSeqNode As XmlNode
Dim xmlPerformernode As XmlNodeList
Dim xmlArtifactListNode As XmlNodeList
Dim xmlTransitionListNode As XmlNodeList
Dim xmlDataAssosListNode As XmlNodeList
Dim xmlDataStoreRefNode As XmlNodeList
Dim xmlDataStoreListNode As XmlNodeList
Dim xmlAssociationListNode As XmlNodeList
Dim xmlMessageFlowListNode As XmlNodeList
Dim eventNode As XmlNode
Dim routNode As XmlNode
Dim gateWay As XmlNode
Dim DataObjects As XmlNodeList
Dim intermediateevenet As XmlNode
Dim i As Integer
Dim z As Integer
Dim s As Integer
Dim performerID As String
Dim artifactID As String
Dim Outsequence As String
Dim performername As String
Dim usecaseID As String
Dim usecasename As String
Dim gateWayType As String
Dim artifactType As String
Dim annotaion As String
Dim messageflowin As String
Dim messageflowout As String
Dim postcondition As String
Dim fs As New FileStream("/data/nobelpz.xpd1", FileMode.Open, FileAccess.Read)
xmlDoc.Load(fs)
xmlNode = xmlDoc.GetElementsByTagName("Activity")
xmlPerformernode = xmlDoc.GetElementsByTagName("Participant")
xmlArtifactListNode = xmlDoc.GetElementsByTagName("Artifact")
xmlTransitionListNode = xmlDoc.GetElementsByTagName("Transition")
xmlDataAssosListNode = xmlDoc.GetElementsByTagName("DataAssociation")
xmlDataStoreRefNode = xmlDoc.GetElementsByTagName("DataStoreReference")
xmlDataStoreListNode = xmlDoc.GetElementsByTagName("DataStore")
DataObjects = xmlDoc.GetElementsByTagName("DataObject")
xmlAssociationListNode = xmlDoc.GetElementsByTagName("Association")
xmlMessageFlowListNode = xmlDoc.GetElementsByTagName("MessageFlow")

'start
HttpContext.Current.Response.Clear()
HttpContext.Current.Response.Charset = ""
```

```
HttpContext.Current.Response.ContentType = "application/msword"

Dim strFileName As String = "GenerateDocument" & ".doc"
HttpContext.Current.Response.AddHeader("Content-Disposition", "inline;filename=" + strFileName)
Dim trigger As String
Dim performer As String
Dim sequence As String
Dim precondition As String
Dim datasourceRef As String
Dim datastore As String
Dim datastoreout As String

Dim strHTMLContent As New StringBuilder()
For k = 0 To xmlNode.Count - 1
    trigger = ""
    performer = ""
    sequence = ""
    precondition = ""
    annotation = ""
    sequence = ""
    datastore = ""
    datastoreout = ""
    messageflowin = ""
    messageflowout = ""
    postcondition = ""
    usecasename = ""
    usecaseID = xmlNode(k).Attributes("Id").InnerText.Trim()
    usecasename = xmlNode(k).Attributes("Name").InnerText.Trim()

    'we need to get list of performers for each activities
    If Not IsNothing(xmlNode(k).Item("Performers")) Then

        xmlchildnode = xmlNode(k).Item("Performers")
        If xmlchildnode.HasChildNodes() Then
            For c = 0 To xmlchildnode.ChildNodes.Count - 1

                performerID = xmlchildnode.ChildNodes(c).InnerText.Trim()

                If performerID <> "" Then
                    usecaseID = xmlNode(k).Attributes("Id").InnerText.Trim()
                    usecasename = xmlNode(k).Attributes("Name").InnerText.Trim()
                    If usecasename <> "" Then

                        For j = 0 To xmlPerformernode.Count - 1
                            If performerID = xmlPerformernode(j).Attributes("Id").InnerText.Trim() Then
                                performername = xmlPerformernode(j).Attributes("Name").InnerText.Trim()
                                If performer <> "" Then
                                    performer = performer & " , " & performername
                                Else
                                    performer = performername
                                End If
                            End If
                        Next
                    End If
                End If
            End If
        End If
    End If
Next
End If
```

```

        Next

    End If
    '
End If
'' end of performers

'first we must get Association

For j = 0 To xmlAssociationListNode.Count - 1
    If usecaseID = xmlAssociationListNode(j).Attributes("Source").InnerText.Trim() Then
        artifactID = xmlAssociationListNode(j).Attributes("Target").InnerText.Trim()
        For m = 0 To xmlArtifactListNode.Count - 1

            If artifactID = xmlArtifactListNode(m).Attributes("Id").InnerText.Trim() Then

                artifactType = xmlArtifactListNode(m).Attributes("ArtifactType").InnerText.Trim()
                If artifactType = "Annotation" Then
                    annotaion = xmlArtifactListNode(m).Attributes("TextAnnotation").InnerText.Trim()
                End If

                ' performerID = xmlPerfnode.InnerText

            End If
        Next
    End If
Next

'end of get Association

'first we must get input data store BPMN2.2
If Not IsNothing(xmlNode(k).Item("InputSets")) Then
    xmldatainputnode = xmlNode(k).Item("InputSets")
    If xmldatainputnode.HasChildNodes() Then

        xmldataSeqnode = xmldatainputnode.Item("InputSet")
        If (xmldataSeqnode.HasChildNodes()) Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes

                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListNode.Count - 1

                        If artifactID = xmlDataAssosListNode(j).Attributes("To").InnerText.Trim() Then

                            Dim dataRefid As String =
                                xmlDataAssosListNode(j).Attributes("From").InnerText.Trim()

                            For m = 0 To xmlDataStoreRefnode.Count - 1
                                If dataRefid = xmlDataStoreRefnode(m).Attributes("Id").InnerText.Trim() Then

                                    Dim dataref As String =
                                        xmlDataStoreRefnode(m).Attributes("DataStoreRef").InnerText.Trim()
                                    For s = 0 To xmlDataStoreListNode.Count - 1
                                        If dataref =
                                            xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() Then

```

```
sequence = sequence & " Reads information from " &
xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim() & "."
' MsgBox("sequence" & sequence)
End If
Next
End If
Next
End If
Next
End If
Next
End If
End If
'end of get inputset BPMN2.2

' sequence output data store BPMN2.2
If Not IsNothing(xmlnode(k).Item("OutputSets")) Then
    xmldatainputnode = xmlnode(k).Item("OutputSets")
    If xmldatainputnode.HasChildNodes() Then
        xmldataSeqnode = xmldatainputnode.Item("OutputSet")
        If xmldataSeqnode.HasChildNodes() Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes
                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListNode.Count - 1

                        If artifactID = xmlDataAssosListNode(j).Attributes("From").InnerText.Trim() Then

                            Dim dataRefid As String =
xmlDataAssosListNode(j).Attributes("To").InnerText.Trim()

                            For m = 0 To xmlDataStoreRefnode.Count - 1
                                If dataRefid = xmlDataStoreRefnode(m).Attributes("Id").InnerText.Trim() Then

                                    Dim dataref As String =
xmlDataStoreRefnode(m).Attributes("DataStoreRef").InnerText.Trim()
                                    For s = 0 To xmlDataStoreListNode.Count - 1
                                        If dataref =
xmlDataStoreListNode(s).Attributes("Id").InnerText.Trim() Then
                                            sequence = sequence & " Writes information on " &
xmlDataStoreListNode(s).Attributes("Name").InnerText.Trim()
                                        End If
                                    Next
                                End If
                            Next
                        End If
                    Next
                End If
            Next
        End If
    Next
End If
End If
End If
```

```
        End If
    ,
End If
' end sequence output data BPMN2.2

'first we must get input data set BPMN2.2
If Not IsNothing(xmlnode(k).Item("InputSets")) Then
    xmldatainputnode = xmlnode(k).Item("InputSets")
    If xmldatainputnode.HasChildNodes() Then

        xmldataSeqnode = xmldatainputnode.Item("InputSet")
        If (xmldataSeqnode.HasChildNodes()) Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes

                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListnode.Count - 1

                        If artifactID = xmlDataAssosListnode(j).Attributes("To").InnerText.Trim() Then

                            Dim dataRefid As String =
                                xmlDataAssosListnode(j).Attributes("From").InnerText.Trim()

                            For z = 0 To DataObjects.Count - 1

                                If dataRefid = DataObjects(z).Attributes("Id").InnerText.Trim() Then
                                    datastore = datastore & " Receive " &
                                        DataObjects(z).Attributes("Name").InnerText.Trim() & ". "
                                End If
                            Next
                        Else
                            End If
                        Next
                    End If
                Next
            End If
        End If

        End If
    End If
'end of get inputset BPMN2.2

' sequence output data set BPMN2.2

'message flow

Dim source As String
For j = 0 To xmlMessageFlowlistNode.Count - 1
    If usecaseID = xmlMessageFlowlistNode(j).Attributes("Target").InnerText.Trim() Then
        source = xmlMessageFlowlistNode(j).Attributes("Source").InnerText.Trim()

        For g As Integer = 0 To xmlnode.Count - 1
            ' performerID = xmlPerfnode.InnerText
            If source = xmlnode(g).Attributes("Id").InnerText.Trim() Then
```



```

        If Not IsNothing(xmlnode(g).Item("Performers")) Then

            xmlchildnode = xmlnode(g).Item("Performers")
            If xmlchildnode.HasChildNodes() Then
                xmlPerfnode = xmlchildnode.Item("Performer")
                performerID = xmlPerfnode.InnerText
                For m = 0 To xmlPerformernode.Count - 1
                    If performerID = xmlPerformernode(m).Attributes("Id").InnerText.Trim() Then
                        performer = xmlPerformernode(m).Attributes("Name").InnerText.Trim()
                        messageflowIN = "Receive " &
xmlMessageFlowlistNode(j).Attributes("Name").InnerText.Trim() & " from " & performer
                    End If
                Next
            End If

        End If

        Next
    ElseIf usecaseID = xmlMessageFlowlistNode(j).Attributes("Source").InnerText.Trim() Then
        source = xmlMessageFlowlistNode(j).Attributes("Target").InnerText.Trim()
        Dim performerofMessage As String
        For g As Integer = 0 To xmlnode.Count - 1
            ' performerID = xmlPerfnode.InnerText
            If source = xmlnode(g).Attributes("Id").InnerText.Trim() Then
                If Not IsNothing(xmlnode(g).Item("Performers")) Then

                    xmlchildnode = xmlnode(g).Item("Performers")
                    If xmlchildnode.HasChildNodes() Then
                        xmlPerfnode = xmlchildnode.Item("Performer")
                        performerID = xmlPerfnode.InnerText
                        For m = 0 To xmlPerformernode.Count - 1
                            If performerID = xmlPerformernode(m).Attributes("Id").InnerText.Trim() Then
                                performerofMessage = xmlPerformernode(m).Attributes("Name").InnerText.Trim()
                                messageflowout = "Send " &
xmlMessageFlowlistNode(j).Attributes("Name").InnerText.Trim() & " To " & performerofMessage
                            End If
                        Next
                    End If
                End If
            End If

        Next

    End If

Next
'end message flow

'' sequence output data
If Not IsNothing(xmlnode(k).Item("OutputSets")) Then
    xmldatainputnode = xmlnode(k).Item("OutputSets")
    If xmldatainputnode.HasChildNodes() Then
        xmldataSeqnode = xmldatainputnode.Item("OutputSet")
    End If
End If

```

```

        If xmldataSeqnode.HasChildNodes() Then
            For Each node As XmlNode In xmldataSeqnode.ChildNodes
                artifactID = node.Attributes("ArtifactId").InnerText.Trim()
                If artifactID <> "" Then
                    For j = 0 To xmlDataAssosListNode.Count - 1

                        If artifactID = xmlDataAssosListNode(j).Attributes("From").InnerText.Trim() Then

                            Dim dataRefid As String =
                                xmlDataAssosListNode(j).Attributes("To").InnerText.Trim()

                            For z = 0 To DataObjects.Count - 1

                                If dataRefid = DataObjects(z).Attributes("Id").InnerText.Trim() Then
                                    datastoreout = datastoreout & "Send " &
                                        DataObjects(z).Attributes("Name").InnerText.Trim() & ". "
                                End If
                            Next
                        Else
                            End If
                        Next
                    End If
                Next
            End If

            '
        End If
    '' end sequence
    'start event

    'IntermediateEvent BPMN2.2

    ' end IntermediateEvent BPMN2.2

    If Not IsNothing(xmlnode(k).Item("Event")) Then

        eventNode = xmlnode(k).Item("Event")
        If eventNode.HasChildNodes() Then
            If Not IsNothing(eventNode.Item("StartEvent")) Or IsNothing(eventNode.Item("IntermediateEvent"))

Then
                trigger = "start"
                usecasename = ""
                '    MsgBox(trigger)
            End If
        End If
    End If
    '
    'end of start event

    'start gateWay

    If Not IsNothing(xmlnode(k).Item("Route")) Then

        usecasename = ""

    ,

```

```
End If
If Not IsNothing(xmlnode(k).Item("Event")) Then

    usecasename = ""

    '
End If
eventNode = xmlnode(k).Item("Implementation")
If Not IsNothing(eventNode) Then

    If eventNode.HasChildNodes() Then

        Dim tasknode As XmlNode = eventNode.Item("Task")
        If tasknode.HasChildNodes() Then
            If Not IsNothing(tasknode.Item("TaskService")) Then
                performer = "System"
            End If
        End If
    End If
End If
End If

'end of gateWay

' transition st get start event and entermediate
' Dim source As String
For j = 0 To xmlTransitionListnode.Count - 1
    If usecaseID = xmlTransitionListnode(j).Attributes("To").InnerText.Trim() Then
        source = xmlTransitionListnode(j).Attributes("From").InnerText.Trim()
        For g As Integer = 0 To xmlnode.Count - 1
            ' performerID = xmlPerfnode.InnerText
            If source = xmlnode(g).Attributes("Id").InnerText.Trim() Then
                If Not IsNothing(xmlnode(g).Item("Event")) Then
                    'start event

                    eventNode = xmlnode(g).Item("Event")
                    If eventNode.HasChildNodes() Then

                        If Not IsNothing(eventNode.Item("StartEvent")) Then

                            trigger = "The Event " & xmlnode(g).Attributes("Name").InnerText.Trim() & "
occurred"

                            ElseIf eventNode.FirstChild.Name = "StartEvent" Then

                                trigger = trigger & "The Event " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " occurred"
                                End If
                                ' Dim startnode As XmlNode = eventNode.FirstChild()

                                End If

                                ' entermediat event
                                If eventNode.HasChildNodes() Then
                                    If Not IsNothing(eventNode.Item("IntermediateEvent")) Then
                                        'here
                                        intermediateevenet = eventNode.Item("IntermediateEvent")
                                        If intermediateevenet.HasChildNodes() Then
```

```

        If Not IsNothing(intermediateevenet.Item("TriggerTimer")) Then
            trigger = trigger & "The time-date " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " is reached."
        End If
        If Not IsNothing(intermediateevenet.Item("TriggerResultSignal")) Then
            trigger = trigger & "The Message " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " Arrives "

        End If
        If Not IsNothing(intermediateevenet.Item("TriggerResultMessage")) Then
            Dim gatewayID = xmlnode(g).Attributes("Id").InnerText.Trim()
            For m = 0 To xmlTransitionListnode.Count - 1

                If gatewayID =
xmlTransitionListnode(m).Attributes("To").InnerText.Trim() Then
                    Dim sourceActivity =
xmlTransitionListnode(m).Attributes("From").InnerText.Trim()

                    For c As Integer = 0 To xmlnode.Count - 1
                        If sourceActivity =
xmlnode(c).Attributes("Id").InnerText.Trim() Then

                            trigger = trigger & "The Message " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " Arrive from " & xmlnode(c).Attributes("Name").InnerText.Trim()

                        End If
                    Next
                End If
            Next

        End If
    Else
        Dim intermediateElement As XmlElement
        intermediateElement = eventNode.Item("IntermediateEvent")
        If (intermediateElement.HasAttribute("Trigger")) Then
            Dim triggertype = intermediateElement.GetAttribute("Trigger")
            If triggertype = "None" Then
                trigger = "The event " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " occurs. "
            End If
        End If
    End If

    ElseIf Not IsNothing(xmlnode(g).Item("Route")) Then

        Dim routeElement As XmlElement = xmlnode(g).Item("Route")
        'we need to determine the type of gateway: split or merg
        Dim gatewayID = xmlnode(g).Attributes("Id").InnerText.Trim()

```

```
Dim INtransitionCoun As Integer = 0
Dim OuttransitionCoun As Integer = 0
For m = 0 To xmlTransitionListNode.Count - 1
    If gatewayID = xmlTransitionListNode(m).Attributes("To").InnerText.Trim() Then
        INtransitionCoun += 1

    End If
    If gatewayID = xmlTransitionListNode(m).Attributes("From").InnerText.Trim() Then
        OuttransitionCoun += 1
    End If
Next
If (routeElement.HasAttribute("GatewayType")) Then
    gateWayType = routeElement.GetAttribute("GatewayType")

    If gateWayType = "Parallel" Then

        Dim nodecount As Integer = 0
        For m = 0 To xmlTransitionListNode.Count - 1

            If gatewayID = xmlTransitionListNode(m).Attributes("To").InnerText.Trim()

Then

                Dim sourceActivity =
xmlTransitionListNode(m).Attributes("From").InnerText.Trim()

                For c As Integer = 0 To xmlnode.Count - 1
                    If sourceActivity = xmlnode(c).Attributes("Id").InnerText.Trim()

Then

                        'check if source is event or gateway then skip
                        If Not IsNothing(xmlnode(c).Item("Route")) Then
                        Else
                            If nodecount = 0 Then
                                precondition = precondition & "The " &
xmlnode(c).Attributes("Name").InnerText.Trim() & " " & "has been completed"
                            Else
                                precondition = precondition & " And The " &
xmlnode(c).Attributes("Name").InnerText.Trim() & " " & "has been completed"
                            End If
                        End If

                        nodecount = nodecount + 1
                    End If
                Next
            End If
        Next

    ElseIf gateWayType = "Inclusive" Then

        If OuttransitionCoun > 1 Then
            precondition = "The " &
xmlTransitionListNode(j).Attributes("Name").InnerText.Trim() & " is True"
        ElseIf INtransitionCoun > 1 Then

            precondition = "The "
            Dim nodecount As Integer = 0
            For m = 0 To xmlTransitionListNode.Count - 1

                If gatewayID =
xmlTransitionListNode(m).Attributes("To").InnerText.Trim() Then
```

```
Dim sourceActivity =
xmlTransitionListNode(m).Attributes("From").InnerText.Trim()

For c As Integer = 0 To xmlnode.Count - 1
    If sourceActivity = xmlnode(c).Attributes("Id").InnerText.Trim()

Then

        If nodecount = 0 Then
            precondition = precondition &

xmlnode(c).Attributes("Name").InnerText.Trim() & " "

        Else
            precondition = precondition & " or " &

xmlnode(c).Attributes("Name").InnerText.Trim() & " "

        End If
        nodecount = nodecount + 1
    End If
Next
End If
Next
precondition = precondition & "has been completed"
End If

ElseIf gatewayType = "Complex" Then

    If OuttransitionCoun > 1 Then
        precondition = precondition & " The " &

xmlTransitionListNode(j).Attributes("Name").InnerText.Trim() & " is True"
    ElseIf INtransitionCoun > 1 Then

        precondition = "The "
        Dim nodecount As Integer = 0
        For m = 0 To xmlTransitionListNode.Count - 1

            If gatewayID =

xmlTransitionListNode(m).Attributes("To").InnerText.Trim() Then
                Dim sourceActivity =

xmlTransitionListNode(m).Attributes("From").InnerText.Trim()

                For c As Integer = 0 To xmlnode.Count - 1
                    If sourceActivity = xmlnode(c).Attributes("Id").InnerText.Trim()

Then

                        If nodecount = 0 Then
                            precondition = precondition &

xmlnode(c).Attributes("Name").InnerText.Trim() & " "

                        Else
                            precondition = precondition & " or " &

xmlnode(c).Attributes("Name").InnerText.Trim() & " "

                        End If
                        nodecount = nodecount + 1
                    End If
                Next
            End If
        Next
        precondition = precondition & "has been completed"
    End If
```

```

        End If
    Else

        If OuttransitionCoun > 1 Then
            precondition = precondition & " The " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " is " & xmlTransitionListNode(j).Attributes("Name").InnerText.Trim()
            'extend relation
        ElseIf INtransitionCoun > 1 Then

            ' precondition = precondition & "The "
            Dim nodecount As Integer = 0
            For m = 0 To xmlTransitionListNode.Count - 1

                If gatewayID = xmlTransitionListNode(m).Attributes("To").InnerText.Trim()

Then
                    Dim sourceActivity =
xmlTransitionListNode(m).Attributes("From").InnerText.Trim()

                    For c As Integer = 0 To xmlnode.Count - 1
                        If sourceActivity = xmlnode(c).Attributes("Id").InnerText.Trim()

Then
                            If Not IsNothing(xmlnode(c).Item("Route")) Then
                                If nodecount = 0 Then
                                    precondition = " The " & precondition &
xmlnode(c).Attributes("Name").InnerText.Trim() & " " & "has been completed."
                                Else
                                    precondition = precondition & " exclusive or " &
xmlnode(c).Attributes("Name").InnerText.Trim() & " " & "Is" &
xmlTransitionListNode(m).Attributes("Name").InnerText.Trim()
                                End If
                            Else
                                If nodecount = 0 Then
                                    precondition = " The " & precondition &
xmlnode(c).Attributes("Name").InnerText.Trim() & " " & "has been completed."
                                Else
                                    precondition = precondition & " exclusive or " &
xmlnode(c).Attributes("Name").InnerText.Trim() & " " & "has been completed."
                                End If
                            End If

                            nodecount = nodecount + 1
                        End If
                    Next
                End If
            Next
            ' precondition = precondition & "has been completed"
            'include relation
        End If

    End If

Else

```

```
preCondition = precondition & "The " & xmlnode(g).Attributes("Name").InnerText.Trim() &
" has been completed."

        End If
    End If
Next

    End If
Next

' end transition

''''''get post condition
' transition st get start event and entermediate
' Dim source As String
For j = 0 To xmlTransitionListNode.Count - 1
    If usecaseID = xmlTransitionListNode(j).Attributes("From").InnerText.Trim() Then
        source = xmlTransitionListNode(j).Attributes("To").InnerText.Trim()
        For g As Integer = 0 To xmlnode.Count - 1
            ' performerID = xmlPerfnode.InnerText
            If source = xmlnode(g).Attributes("Id").InnerText.Trim() Then
                If Not IsNothing(xmlnode(g).Item("Event")) Then
                    ''start event

                    eventNode = xmlnode(g).Item("Event")

                    'entermediat event
                    If eventNode.HasChildNodes() Then
                        If Not IsNothing(eventNode.Item("EndEvent")) Then
                            If xmlnode(g).Attributes("Name").InnerText.Trim() <> "" Then
                                postcondition = postcondition & "The " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " is created" & ". The process ends."
                            Else
                                postcondition = postcondition & "The process ends"
                            End If
                        End If
                        If Not IsNothing(eventNode.Item("IntermediateEvent")) Then
                            'here
                            intermedieevenet = eventNode.Item("IntermediateEvent")
                            If intermedieevenet.HasChildNodes() Then
                                If Not IsNothing(intermedieevenet.Item("TriggerTimer")) Then
                                    postcondition = postcondition & "The time-date " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " event is created."

                                End If
                                If Not IsNothing(intermedieevenet.Item("TriggerResultSignal")) Then
                                    postcondition = postcondition & "The Signal " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " is send."

                                End If
                                If Not IsNothing(intermedieevenet.Item("TriggerResultMessage")) Then
                                    Dim gatewayID = xmlnode(g).Attributes("Id").InnerText.Trim()
                                    postcondition = postcondition & "The Message " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " is send. "

                                End If
                            End If
                        End If
                    End If
                End If
            End If
        Next
    End If
Next
End If
```



```
Else
    Dim intermediateElement As XmlElement
    intermediateElement = eventNode.Item("IntermediateEvent")
    If (intermediateElement.HasAttribute("Trigger")) Then
        Dim triggertype = intermediateElement.GetAttribute("Trigger")
        If triggertype = "None" Then
            postcondition = postcondition & "The event " &
xmlnode(g).Attributes("Name").InnerText.Trim() & " occurs. "

            End If
        End If
    End If

    End If
End If

Else
    'preCondition = preCondition & "The " & xmlnode(g).Attributes("Name").InnerText.Trim() &
" has been completed."

    End If
End If
Next

End If
Next

'get workflow name for actors
Dim activitiParent As XmlNode = xmlnode(k).ParentNode()
Dim workflow As XmlNode = activitiParent.ParentNode()
' performer = workflow.Attributes("Name").InnerText.Trim()
'end workflow
' begin datastore get it from DataAssociation
For j = 0 To xmlDataAssosListNode.Count - 1
    If usecaseID = xmlDataAssosListNode(j).Attributes("To").InnerText.Trim() Then
        source = xmlDataAssosListNode(j).Attributes("From").InnerText.Trim()
        For g As Integer = 1 To xmlDataStoreRefnode.Count - 1
            If source = xmlDataStoreRefnode(g).Attributes("Id").InnerText.Trim() Then
                datasourcesRef = xmlDataStoreRefnode(g).Attributes("DataStoreRef").InnerText.Trim()
                For c As Integer = 1 To xmlDataStoreListNode.Count - 1
                    If datasourcesRef = xmlDataStoreListNode(c).Attributes("Id").InnerText.Trim() Then
                        datastore = xmlDataStoreListNode(c).Attributes("Name").InnerText.Trim()
                    End If
                Next
            End If
        Next
    End If
Next

End If
Next

'end datastore

'start writing
If usecasename <> "" Then
```

```
' MsgBox(trigger)
strHTMLContent.Append("<br>".ToString())
strHTMLContent.Append("<table align='Left' height='500px' style='border:1px solid black'>".ToString())
strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid black'><b>Use Case
Name</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" &
usecasename & "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())

strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid black'><b>Actors</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" &
performer & "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())

strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid black'><b>Trigger</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" & trigger
& "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())

If postcondition <> "" Then
strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid black'><b>Post-
Condition</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" &
postcondition & "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())
End If

strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid black'><b>Pre-
Condition</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" &
preCondition & "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())

strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid black'><b>Scenario</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" &
sequence & "<br>" & messageflowin & "<br>" & Outsequence & "<br>" & datastore & " " & datastoreout & " " &
messageflowout & "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())
If annotaion <> "" Then
strHTMLContent.Append("<tr>".ToString())
strHTMLContent.Append("<td style='width: 180px;border:1px solid
black'><b>Comment</b></td>".ToString())
strHTMLContent.Append("<td style='width: 320px;background:#FFFFFF;border:1px solid black'><b>" &
annotaion & "</b></td>".ToString())
strHTMLContent.Append("</tr>".ToString())
End If

strHTMLContent.Append("</table>".ToString())

strHTMLContent.Append("<br><br>".ToString())
```

```
strHTMLContent.Append("<br><br>".ToString())  
strHTMLContent.Append("<br><br>".ToString())  
strHTMLContent.Append("<br><br>".ToString())  
strHTMLContent.Append("<br><br>".ToString())  
strHTMLContent.Append("<br><br>".ToString())
```

```
End If
```

```
strHTMLContent.Append("<br><br>".ToString())  
strHTMLContent.Append("<br><br>".ToString())
```

```
Next
```

```
HttpContext.Current.Response.Write(strHTMLContent)
```

```
HttpContext.Current.Response.End()
```

```
HttpContext.Current.Response.Flush()
```

```
End Sub
```