City Research Online

City, University of London Institutional Repository

Citation: Parhizkar, M. (2016). Impact analysis of enterprise resource planning post-implementation modifications. (Unpublished Doctoral thesis, City, University of London)

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/16151/

Link to published version:

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

IMPACT ANALYSIS OF ENTERPRISE RESOURCE PLANNING POST-IMPLEMENTATION MODIFICATIONS

A thesis submitted for the degree of
Doctor of Philosophy (PhD) in Software Systems Engineering
City University of London

School of Mathematics, Computer Science and Engineering
Department of Computer Science

By
Minou Parhizkar
2016

Thesis written by
Minou Parhizkar
MSc. Business System Integrating (with SAP Technology),
Brunel University, United Kingdom, 2009
BSc. Computer System Engineering,
Brunel University, United Kingdom, 2008

Supervised by
Doctor Marco Comuzzi
Doctor George Buchanan
Professor Neil Maiden

TABLE OF CONTENTS

LIST OF FIGURES ..... **XI**

LIST OF TABLES ..... **XV**

ACKNOWLEDGEMENTS ..... **XIX**

ABSTRACT ..... **XXI**

CHAPTER 1 INTRODUCTION **23**

1.1 Introduction 23

1.2 Background and Motivation 23

1.3 Research Question and Objectives 26

1.4 Research Methodology 27

1.5 Thesis Structure 29

CHAPTER 2 LITERATURE REVIEW **33**

2.1 Introduction 33

2.2 ERP Systems Concepts and Features 33

2.2.1 ERP Systems Benefits 35

2.2.2 ERP Systems Architecture 36

2.2.3 ERP Systems Implementation Lifecycle 38

2.3 ERP Post-implementation 40

2.3.1 Reason for ERP Systems Change 41

2.3.2 Solutions for Change in Requirements 43

2.3.3 ERP Package Modification 44

2.4 ERP Systems Modification and Change Management 51

2.5 Change Management Terminology 51

2.5.1 Change Management Process 52

2.5.2 Engineering Change Management 54

2.6 Impact Analysis Approach 56

2.6.1 Impact analysis of Enterprise System 58

2.6.2 Impact Analysis of Business Processes 59

2.7 Summary 60

CHAPTER 3 RESEARCH METHODOLOGY **63**

3.1 Overview of Design Science Research 63
3.2 Adaption of Design Science Research 64
3.3 Research Road Map and Phases 66
3.4 Research Validation and Analysis Method 68
3.5 Summary 70
CHAPTER 4 FRAMEWORK DESIGN PART 1 (PROCESS AND CONCEPTUAL DEPENDENCIES) 71
4.1 Introduction 71
4.2 ERP Change Process 72
4.2.1 Phase One: Understanding ERP System Dependencies 75
4.2.2 Phase Two: Requirement Elicitation and Analysis 76
4.2.3 Phase Three: Modification Analysis 77
4.2.4 Phase Four: Modification Implementation 80
4.3 ERP Dependency Meta-Model 82
4.3.1 Run-time and Design-time 86
4.4 Taxonomy of Change in ERP Systems 88
4.4.1 Granularity: Atomic and Composition Change 90
4.4.2 Level and Type of ERP Modification 90
4.5 Change Request Template 93
4.6 Example of Change Scenario 95
4.7 Summary 99
CHAPTER 5 FRAMEWORK DESIGN PART 2 (IMPACT ANALYSIS AND ASSESSMENT) 101
5.1 Introduction 101
5.2 Preliminaries to Design Impact Analysis Mechanisms 102
5.3 Compatibility Among ERP Systems Components 103
5.3.1 Compatibility Assessment Business Data 106
5.3.2 Compatibility Assessment Business Function 109
5.3.3 Compatibility Check Business Process 111
5.4 Critical Point for Running Instances Affected by Change 113

5.4.1 Migration Policies for Running Instances Affected by Change 113
5.4.2 Critical Point in Running Instances Affected by Change 115
5.5 Propagation Mechanism.....120
5.5.1 Impact Analysis Algorithm: Update Algorithm 123
5.5.2 Impact analysis algorithm: Delete Algorithm.....133
5.5.3 Impact analysis algorithm: Add item.....135
5.6 Impact Metrics 136
5.6.1 Level 1: Impact Propagation (Relative Fraction of Impacted Items) 138
5.6.2 Level 2: Relative Weight Assigned to Different Impact Categories 140
5.6.3 Impact Weight Factor 140
5.6.4 Metrics for Estimating the Cost/Effort Implications for Modification Implementation.....143
5.7 Summary.....148
CHAPTER 6 COST IMPLICATION OF ERP MODIFICATION 151
6.1 Introduction.....151
6.2 Classification of Different Strategies for ERP Modification.....152
6.3 AHP Methodology for Decision-Making 154
6.4 Design Empirical Study.....155
6.5 Analysis of the Results.....159
6.5.1 Normalization 159
6.5.2 Consistency Check.....160
6.5.3 Aggregate the Responses.....161
6.6 Discussion.....163
6.7 Summary.....165
CHAPTER 7 PROOF OF CONCEPT IMPLEMENTATION.....167
7.1 Introduction.....167
7.2 Software Development Process: 167
7.3 Requirement Specification.....169
7.3.1 Impact Analysis Functionality and Development Iteration 169
7.4 Design and Implementation Specification.....176

7.4.1 Model Driven Approach 176
7.4.2 Mendix: A Model-Driven Application Development Tool 176
7.4.3 Mendix Features 177
7.4.4 Mapping our Framework to Mendix 180
7.4.5 Implementation of Tool GUI 185
7.5 Tool Demonstration: Running Example Walkthrough 185
7.5.1 Mapping the ERP Dependencies 185
7.5.2 Requirement Analysis 188
7.5.3 Modification Analysis 190
7.6 Summary 197
CHAPTER 8 FRAMEWORK EVALUATION AND DISCUSSION 199
8.1 Introduction 199
8.2 Evaluation Method 199
8.2.1 Evaluation Goal 199
8.2.2 Evaluation Method 201
8.3 Evaluation Phase 1: Case Study Evaluation 203
8.3.1 GBI SAP Case Study 204
8.3.2 CRONUS Microsoft Dynamic NAV 211
8.3.3 Discussion for Phase One Evaluation: 217
8.4 Evaluation Phase 2: User Study 218
8.4.1 ERP Expert Study (Part I) 219
8.4.2 Non-expert Study (Part II) 222
8.4.3 Data analysis 223
8.5 Results 224
8.5.1 ERP Experts Interview Assessment (Part I) 224
8.5.2 Result for functionality assessment with ERP expert (Part II-A) 228
8.5.3 Result for Usability assessment with both group (Part II-B) 233
8.6 Summary 239
CHAPTER 9 CONCLUSIONS 241
9.1 Introduction 241

9.2 Research Summary and Achievement 241
9.3 Contribution 244
9.4 Research Limitation and Future Research 247
9.5 Publication 248
REFERENCES 251
APPENDIX. A IMPACT ALGORITHM.....263
APPENDIX. B QUESTIONNAIRE FIRST STUDY 267
APPENDIX. C QUESTIONNAIRE SECOND STUDY 273
Part A) Study with ERP Expert 273
Part B) Study with Non-Expert 287
APPENDIX. D IMPLEMENTATION MICROFLOW, DOMAIN.....299
APPENDIX. E RESULT OF THE STUDY 311
Result from AHP- Method 311
Interview transcript with ERP professional.....312
Experimental Study with ERP expert.....319
Experimental Study with Non-ERP expert.....326

LIST OF FIGURES

Figure 1-1 DSR framework (Hevner et al. 2004) 28
Figure 2-1 ASAP Roadmap (Gulledge and Simon 2005, Miller 1998) 39
Figure 2-2- Engineering change process (Jarratt 2004) 54
Figure 2-3 Understanding Software Change and determine Impact (Bohner, 2000) 55
Figure 3-1 DSR framework (Hevner et al. 2004) 64
Figure 3-2 DSR framework contextualized to this research (Hevner et al. 2004) 65
Figure 3-3 Research Iteration Phases 67
Figure 4-1 A model of a generic change process from (Jarratt 2004) 73
Figure 4-2 Framework for ERP modification 74
Figure 4-3 Change Process as Phase 1 81
Figure 4-4 Four Tier of ERP Enterprise Model 83
Figure 4-5 ERP Dependency Meta-Model 84
Figure 4-6 Example of Running instances 88
Figure 4-7 Change Process at Phase 2 89
Figure 4-8 Taxonomy of ERP Modification 91
Figure 4-9 Change Process at Phase 2 (Requirement Elicitation) 93
Figure 4-10 Change Process of ERP Modification 96
Figure 4-11 Example of Purchasing Process 97
Figure 5-1 Change Process at Phase 3 102
Figure 5-2 Compatibility assessments 105
Figure 5-3 Example of updating purchase requisition 108
Figure 5-4 Example of Updating Function (Create Purchase Order) 111
Figure 5-5 Example of Updating Business Process 113
Figure 5-6 Example of Critical Point 116
Figure 5-7 Notion of Critical Point for Simple Case 118
Figure 5-8 Notion of Critical Point for Complex Case 119

Figure 5-9 Example of Update Purchase Requisition 128
Figure 5-10 Metrics Level 138
Figure 5-11 Cost Calculation 148
Figure 6-1 Representation of Hierarchical Model 155
Figure 6-2 Hierarchy Model based on configuration and code modification 156
Figure 6-3 Hierarchy Model of overall strategies 157
Figure 6-4 Distribution of ERP Expert 158
Figure 6-5 Example of Questionnaire 159
Figure 6-6 Sample of Matrix result for one 160
Figure 6-7 Consistency Check 161
Figure 6-8 Weight Priorities 162
Figure 6-9 Final Cost Results 163
Figure 7-1 Representation of impact analysis tool based on change process framework 171
Figure 7-2 Use Case of Impact analysis tool 173
Figure 7-3 First Iteration ERP Dependency Model 174
Figure 7-4 Second Iteration Impact analysis 175
Figure 7-5 Third Iteration Cost Implication 176
Figure 7-6 Mendix Domain Model (Example of function and data object) 178
Figure 7-7 Microflow Elements 179
Figure 7-8 Mendix Microflow (Create Business Function) 180
Figure 7-9 Mapping Process (Dependency model to Domain model) 182
Figure 7-10 Mapping Process BPMN to Mendix Microflow (analysis of change request) 183
Figure 7-11 Mapping the BPMN for the algorithms 4 to Microflow 184
Figure 7-12 Example of ERP system 186
Figure 7-13 Example of mapping ERP System components 187
Figure 7-14 Example of Process Mapping of ERP system 188
Figure 7-15 Change Request List 189
Figure 7-16 Example of Change Request 190
Figure 7-17 Example of impact analysis list 191
Figure 7-18 Example of Impact Summary Report 192

Figure 7-19 Example of Impact Assessment Part one (Percentage) 193
Figure 7-20 Example of impact assessment part two (propagation and risk) 194
Figure 7-21 Example of Implementation Strategies 196
Figure 7-22 Example of Cost implication of modification 197
Figure 8-1 GBI Production Process 206
Figure 8-2 GBI Purchasing Process 206
Figure 8-3 GBI Sales and Distribution Process 207
Figure 8-4 Screenshot of mapping items 209
Figure 8-5 Implementation of Change Scenario 209
Figure 8-6 CRONUS Purchase and payable process 212
Figure 8-7 CRONUS Sale and Receivable 212
Figure 8-8 Example of Mapping Business Process of CRONUS 214
Figure 8-9 Example of Change to CRONUS and the Result by Impact Analysis 215
Figure 8-10 Result for missing item 230
Figure 8-11 Result for rating the accuracy of impact analysis 231
Figure 8-12 Result of impact analysis functionality 232
Figure 8-13 Perceived Average Usefulness 234
Figure 8-14 Perceived average ease of use 235
Figure 8-15 Perceived average ease of learn 237
Figure 8-16 Perceived average satisfaction 237

LIST OF TABLES

Table 2-1 ERP Customization Option 46
Table 3-1 Knowledge base of the research 66
Table 4-1 Taxonomy of Business Process Modification (Reichert and Weber 2012) 92
Table 4-2 Change Request Template 94
Table 4-3 Example of Change Request 98
Table 5-1 Set theoretic representation of ERP components 121
Table 5-2 Addition Modification 135
Table 6-1 Implementation Strategies for each type of Change 154
Table 6-2 Pairwise comparison Scale 158
Table 8-1 Total item captured in GBI case study 205
Table 8-2 Comparison of impact analysis tool vs. ERP functionality of SAP 210
Table 8-3 Total items captured in CRONUS case study 211
Table 8-4 Comparison of impact analysis tool vs. ERP functionality of Microsoft Dynamic 216
Table 8-5 Summarized result from all ERP expert 228
Table 8-6 Frequency of Rating Accuracy of Impact Analysis 231
Table 8-7 Frequency of Rating Functionality of Impact Analysis 232
Table 8-8 Perceived Usefulness 234
Table 8-9 Perceived Ease of Use 235
Table 8-10 Perceived Ease of learn 236
Table 8-11 Perceived satisfaction 237
Table 9-1 Interviewee 1 312
Table 9-2 Interviewee 2 313
Table 9-3 Interviewee 3 314
Table 9-4 Interviewee 4 315
Table 9-5 Interviewee 5 316

Table 9-6 Interviewee 6 317
Table 9-7 Interviewee 7 318

DEDICATION

My beloved father, Alireza Parhizkar,
for raising me, teaching me and loving me.

My delightful sister and brother, Maryam and Ehsan Parhizkar,
for being so tremendously patient and supportive.

The loving memories of my Mother, Ferry Khazainezhad,
for being such excellent human being who will always be missed.

ACKNOWLEDGEMENTS

I am deeply indebted to many people who have provided help, support and encouragement throughout my research in this thesis.

First of all, I want to express my gratitude to my supervisor, Dr Marco Comuzzi, who always believed in me, for all the work we have accomplished together and for all invaluable advice, useful criticism, and his amazing ability to find solutions to even the most complex situations. I felt every discussion with him full of new idea and energy towards my research, and I appreciate the amount of time he was willing to dedicate toward my growth as a researcher. I would never have made it through these years without him and cannot imagine having a better supervisor.

Second, I would like to thank my other supervisor Dr George Buchanan for his useful hints, help, and advice. I am grateful for all the support I received from him during my research journey and especially the last few months of my research.

Furthermore, I also would like to thank Ms Stephanie Wilson, for all her support and my co-supervisor Neil Maiden for his advice and feedback and finally special thanks to Dr Simon Stumpf, who introduce me to Dr Marco Comuzzi.

Moreover, I had the pleasure to work together with many great colleagues who accompanied me during the pursuit of my Ph.D. and who also deserve to be mentioned here, Dr Graham Dove, Ms Adrian Bussone, Dr Dara Sherwani, Ms Milena Markova, and Ms Tracey Booth for being the best colleagues and friends and for their endless support and discussion during our lunchtime events.

I would also like to thank all the members of the Centre for Human-Computer Interaction Design and Interaction Lab of City University for all their support and help during the period of my studies.

A special thanks to my father Mr Alireza Parhizkar who encouraged me during the last couple of years, Ms Maryam Parhizkar, and Mr Ehsan Parhizkar, who always been there for me during my study.

I cannot imagine to be able to have this achievement without these important people.

Minou,

London, UK

ABSTRACT

Enterprise Resource Planning (ERP) as business system integration evolves in the post-implementation phase due to the change in business requirements caused by competitive environments. Uncontrolled or poorly managed changes may lead to low quality, chaotic systems and data that are difficult to use and maintain. Constructivist approaches to effectively manage post-implementation change in ERP systems from the design-related standpoint are currently lacking. Research in this field mostly focuses on CSF (Critical Success Factors) of the post-implementation phase rather than providing a well-structured approach for managing the changes. Thus, this thesis is designed to close this gap by devising methods and tools for controlled ERP post-implementation change management to support stakeholders, such as business analysts and developers, in assessing the impact of the modification.

Our methodology draws a parallel approach between ERP post-implementation change management and traditional engineering change management in product design and proposes a framework for impact analysis of ERP post-implementation modifications. The framework defines a meta-model of the dependencies among ERP entities such as business processes, functions, and data. Based on the identified dependencies, the framework allows to automatically analyse the impact of a proposed change through a set of impact analysis mechanisms. Then, evaluate the scope and depth of a proposed change through a set of impact assessment metrics.

As part of the evaluation process, our framework has been embedded in a software tool i.e. decision support system to demonstrate the feasibility of our approach. Then, provides an empirical study to validate the research method and the tool through a panel of ERP experts and end users. The result confirms that our framework provides scientifically grounded method to manage ERP post-implementation modification in a controlled manner. The application of our approach improved change impact analysis and reduced the risk associated with post-implementation change management in future.

CHAPTER 1 INTRODUCTION

1.1 Introduction

This chapter provides an overview of the context of this thesis, explaining the research goal and describing what this thesis contributes to the research on ERP post-implementation and change impact analysis. Moreover, the structure of this thesis is outlined to guide the reader.

1.2 Background and Motivation

The academic literature defines ERP (Enterprise Resource Planning) systems as business applications capable of automating and integrating the organisation's business processes and data in a unified system (Klaus, Rosemann, and Gable 2000). In the last three decades, ERP systems became a solution for most enterprises to manage their data and business processes. The development of custom ERP applications is expensive; therefore, ERP systems are mostly selected as off-the-shelf software packages (Klaus, Rosemann, and Gable 2000). ERP systems are designed as software packages matching the general needs of organisations (Luo and Strong 2004). However, there is often a gap between enterprise requirements and the business functions of the ERP package, which creates a negative impact on the organisation (van Beijsterveld 2006).

Consequently, frequent changes are vital for ERP systems that are deployed in an ever-changing context. A study indicates that companies spent most of their budget, between 50% up to 70%, on the maintenance of their software systems (Bennett 1990). This is due to the complexity of enterprise systems and the gap between the system and the organisational needs.

In order to take full advantage of ERP systems, implementations require drastic structural and cultural changes within the organisation including business process re-evaluation and re-engineering. These changes are difficult to accomplish, and organisations continue to struggle with change management of ERP systems.

Over the years, many different strategies have been developed to support requirements analysis and management during post-implementation phases in enterprise systems. In the post-implementation phase, the misfit between the ERP system and the organisation's requirements can be tackled in three ways: (i) ERP system modification, through either package configuration or ad-hoc customization, (ii) organisational adaptation or (iii) a combination of both (Zach and Erik Munkvold 2012).

ERP systems are modified mainly because of three factors. The adaptive factor applies when the organisation requires improving an ERP system to address new business requirements. The corrective factor concerns modifications to address weaknesses or correct errors identified during system usage. Finally, the perfective factor involves the alignment of the organisation requirements with the ERP system capability. Examples of some of the above reasons can be as follows:

- Changing ERP user needs demand for the addition of new functionality and features;
- Fixing bugs and security issues;
- The requirement to support new hardware devices by ERP systems, e.g. adding touch-based user interaction as a way of payment or fingerprint for authorization.
- Business performance improvements are demanding for optimisation of ERP application.

The maintenance effort has been estimated to be frequently more than 50% of the total life cycle cost (Lee 1998). By identifying potential impacts before making a change, the risks associated with embarking on the effort can be reduced. However, apart, from adding new features, fixing bugs, etc., the change can also have adverse effects on ERP systems. Changes are often accompanied by unintended side effects. These side effects typically result in new bugs or impose new authorization and security problems. Also, the side effects can decrease the maintainability of an ERP system.

Side effects result from changes that were implemented in an inconsistent or incorrect manner, which is mostly caused by unseen dependencies that exist among the components of an ERP system.

Change impact analysis is a strategic approach to analysing the side effects in large software systems. Early research contributed by Bohner and Arnold (Bohner 1996) investigated the foundation of software change impact analysis and provided the following definition of the term impact analysis that has been adopted by most researchers today:

"Identifying the potential consequence of change or estimating what needs to be modified to accomplished a change. "(Bohner 1996) Pg. 3

A number of techniques for analysis of the impact of change in the source code (Chan et al. 2009), workflow systems (Oliva et al. 2013), business processes (Weber, Rinderle, and Reichert 2007a) and service-oriented systems (Wang, Yang, and Zhao 2010) have been discussed in the literature (Sun et al. 2010). These different types of analysis are typically associated with a certain view and perspective of software, such as an architectural view or source code view. Moreover, the views are related to different stakeholders, such as programmers, requirement engineers or system support who are responsible for maintaining the software artefact. While these techniques provide an excellent example of how to apply ripple effect analysis in specific domains, they can be difficult to employ to assess the modification of ERP systems, which are not only complex software systems, but that also have a direct impact on an organisation's business performance. According to (Bohner 1996) Pg. 3 a ripple effect is the "effect caused by making a small change to a system which affects many other parts of the system". Since there is no known existing method of Change Impact Analysis specifically for ERP systems, this research becomes imperative.

Additionally, most impact analysis approaches do not distinguish between the effects of different types of changes at design-time and run-time. The majority of the works related to impact analysis are focused on the impact of modifications at design-time, i.e., the "build-time" of software artefacts. ERP systems are complex software systems supporting business operations that may be long lasting. This means that change impact analysis should not only focus on the design-time structure of the system but should also address the impact of changes in ongoing business operations, such as long-running instances of business processes that are not yet completed during the assessment of ERP change.

As a summary, ERP post-implementation changes bear an impact on both the static design-time structure of ERP systems and the run-time instances currently executing.

Technique and tools for controlled change management in this context should concern both the design-time and run-time domains and support the business analyst in implementing the change in the smoothest way possible for the ERP system and consequently, for the organisation.

This thesis proposes a framework, i.e., methods and tools, to support change management in the ERP post-implementation phase. The proposed framework can adequately address different types of changes in ERP systems, to assist business analysts to understand and retrace the ripple effects of a proposed change in an ERP system and support in determining the most efficient plan for implementation of the modification.

1.3 Research Question and Objectives

The main goal of this thesis with the problem statement of the previous section is to develop techniques for controlling change management of modifications of the ERP system in the post-implementation phase.

The main research question addressed by this work is:

"*How is it possible to design a framework to support the management, i.e., specification, analysis, and assessment, of ERP post-implementation changes?"*

More specifically, our approach aims at the following four objectives:

1. Develop a generic conceptual meta-model of ERP systems to determine the dependencies among the different components constituting the system; This meta-model could then be instantiated into a specific model of dependencies among components in specific ERP installations;
2. Introduce a taxonomy of possible post-implementation modifications of ERP systems, based on the dependencies defined by the conceptual meta-model;
3. Define a methodology to assess the impact of different types of change, by considering, in particular, the ripple effects implied by specific dependencies;
4. Define metrics to estimate the depth of the impact of ERP post-implementation change, possibly based on the strategy selected to implement the identified change.

a. As a part of impact assessment gathering from ERP expert about the relative cost of alternative strategies for implementation of modification using an AHP-based method.

5. Implement a software tool, i.e., a decision support system, embodying the identified models, methods and metrics to assist business analysts in the controlled management of ERP post-implementation change.

1.4 Research Methodology

In general, IS research follows one of two main research strategies: Constructivist or behavioural research. The behavioural research is to understand and predict real-world phenomena while constructivist research aims at solving practical and theoretical organisational problems by developing and evaluating IT artefacts (Winter 2008). By analysing the characteristics associated with qualitative, quantitative, inductive, and deductive research methods and mapping them against the attributes of this research, the appropriate research approach for this work is, determined.

This research need is driven by the difficulties encountered by organisations in managing ERP post-implementation change, as highlighted in Section 1.2. Thus, it is considered a problem-oriented approach such as what is proposed by (Hevner et al. 2004). According to (Hevner et al. 2004), Pg. 78 "design science creates and evaluates IT artefacts aimed at solving identified organisational problems". It is involved with structuring design research and driving it through from empirical studies of developing the introduction of novel methods and tools to support the improvement of the design process. Design science research is, in particular, effective for formulation and validation of techniques and tools since it incorporates clearly defined criteria of success measurement.

In this thesis applying the design science research methodology as a framework provides the researcher to validate knowledge systematically, and at the same time to ensure that the research is scientific and delivers valid results.

The (Hevner et al. 2004) methodology suggests that design research should draw design problems from both existing literature and design practice. The process model consists of six constructs representing activities that should be carried out during a Design

Science research. The activities include problem identification and motivation; define the objectives for a solution; design and the development; demonstration; evaluation; and communication. Chapter 3 explains each these activities in detail.

As described in Figure 1-1, design science is tied to the environment as a source for relevance and the existing knowledge base as a source for rigor. The design science artefact is developed and evaluated in the design science cycle. The environment contributes the application domain of the design artefact. Within the application domain, people, organisational systems, and technical systems provide the context for the research. Also, problems and opportunities originate from the application domain and can be translated into requirements for the design science artefact.

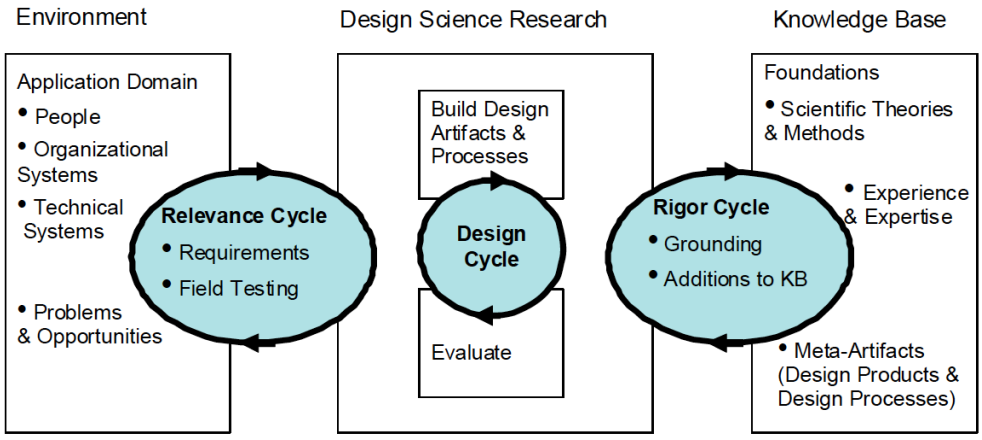


Figure 1-1 DSR framework (Hevner et al. 2004)

As mentioned before this thesis focused on the design of a framework for impact analysis of modifications of ERP systems during post-implementation to assess and evaluate the change. In order to provide the solution for managing ERP post-implementation change to achieve our objectives delineated in Section 1.3, this research work develops and evaluates the following artefacts:

- A methodology to identify the different phases of controlled ERP post-implementation change management;
- A dependency meta-model of ERP system components that can be applicable for mapping dependency relations;

- A taxonomy of possible ERP post-implementation changes;
- A set of mechanisms (algorithms) to evaluate the ripple effects of the proposed ERP changes, based on the dependencies identified by the dependency meta-model;
- A set of business intelligence metrics to assess the impact of the proposed modification;
- A software tool (decision support system) embedding all the conceptual artefacts defined above, to assist business analysts in change management.

In addition, the proof-of-concept implementation of the software tool can be considered as the evaluation to demonstrate the capability of implementation of our proposed methods. During development of the tool, an agile methodology and a model-driven approach were chosen as the foundation of our implementation. Following the design and implementation of a software tool, this thesis explores the viability of the discussed concepts and approach and plays a formative role for further development. Further, this research is designed to provide empirical evidence to validate the achievement of our objectives. The main purpose of evaluation is to show that our design solution has certain properties that work under certain condition and behave in a particular way that can be useful for the ERP specialist and facilitates a solution during the change process. We evaluated our approach, with a panel of ERP experts, i.e., business analysts and solution designers from large consulting companies. Also, we ran controlled experiments in simplified scenarios using former ERP students and ERP professional as participants to evaluate the usability of our designed software tool.

1.5 Thesis Structure

The remainder of the thesis at hand is organized as follows.

Chapter 2 (Literature Review): This chapter describes the concepts used in the thesis and explores the state of the art of current research on ERP system post-implementation, ERP customization, change management and various techniques and approaches that have been proposed for managing the change at the post-implementation phase of ERP system. Furthermore, the chapter presents and discusses the result of a systematic literature review and taxonomy of impact analysis that was driven from existing approach used in software

systems. Lastly, the chapter discusses in detail the existing approaches used for assessing modifications in enterprise systems and business process management, to underline the identified gap in the literature.

Chapter 3 (Research Methodology): In this chapter, the design science approach is introduced then, presented the structure of the proposed methodology and adaptation to the research problem. In the sub-section of this chapter, the research roadmap is defined which explains phases of the research approach and iterations of the research. The chapter ended by describing a brief summary of research evaluation and analysis method for research validation.

Chapter 4 (Framework Design Part 1 process and conceptual dependencies): In this chapter, we present our framework to support the assessment and evaluation of change impact analysis of ERP systems. This chapter reports the overall change process of our framework and presents a comprehensive analysis of ERP dependency relationships and ERP change requirements analysis. First, a standard change process for ERP systems modifications is presented. Then, we present an overview of the artefact involved in the change process. Finally, this chapter presents in detail a first subset of the artefacts defined in our framework, namely the ERP dependency meta-model artefact for mapping the dependency relationship of ERP components, the taxonomy of ERP modifications and the ERP change request template.

Chapter 5 (Framework Design Part 2 impact analysis and assessment): This chapter completes the description of our framework, by focusing on the issues of change impact analysis and assessment. First, it describes the impact analysis mechanisms, which capture the ripple effects of ERP modifications on the existing design-time and run-time structure of the ERP system. Then, we present of metrics for assessing the impact of a proposed ERP change. These metrics aim at enhancing the decision making to plan the implementation of the modification efficiently.

Chapter 6 (Cost Implication of ERP Modification): This chapter focuses on the issue of evaluating the relative cost of different strategies for the implementation of ERP modifications. In particular, it provides more detail on the parameterization of functions to estimate the implementation effort of ERP changes. The results presented in this chapter

have been used to improve the impact assessment metrics in the software tool implementation. The estimation of effort function relies on ERP experts' opinions, and it is built on the AHP (Analytic Hierarchical Process) multi-attribute decision-making techniques.

Chapter 7 (Proof of Concepts implementation): This chapter gives more insight into the design and implementation of the impact analysis tool as proof of concept. First, it provides a description of the software development methodologies and explains the reasons for choosing an agile methodology for implementation of impact analysis. Then it provides a detail description of the features and functionality of impact analysis and discusses the design and implementation of the artefact development through adaptation of model-driven approach. The chapter concludes by providing an example of a case in order to test and demonstrate the functionality and feasibility of tool implementation.

Chapter 8 (Framework Evaluation and Discussion): This chapter provides empirical evidence to validate our approach and impact analysis tool. It explains the evaluations goal, evaluation criteria, and the evaluation method in order to achieve our objective. In this chapter two ERP case studies demonstrate and present the result for perceiving the feasibility of applying impact analysis tool. Then we evaluate the applicability and functionality of our approach and tools through the study with ERP expert. Finally, we assess the usability of our tool by conducting an experimental study in two groups of ERP experts and non-experts (students) in order to evaluate the usefulness, ease of use and satisfaction of the application of impact analysis tool.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction

This chapter investigates the state of the art of current research on ERP system post-implementation and the change impact analysis field based on the objectives of this thesis. We report on the finding of a comprehensive literature study and analyse various techniques and approaches that have been proposed for managing the change at the post-implementation phase of the ERP system. The chapter outlines a detailed overview of ERP system concepts and architecture in Section 2.2 then explains the issue of ERP customization during the post-implementation in Section 2.3. As change management is often identified as a fundamental critical success factor of ERP post-implementation, Section 2.4 explores this principle and presents the change management terminology and procedures in, Section 2.5, which uses a strategic approach for handling the change. Then Section 2.6 discusses the impact analysis concepts for a software system and explores the existing methods and techniques that used as activities during the change management process. This section also outlines the strength and weaknesses of the proposed approach in regards to ERP system modification. Moreover, this chapter summarises open research problems and prepares the ground for refining the research goal of this thesis.

2.2 ERP Systems Concepts and Features

From the perspective of (Iivari 1991), an Information System is a combination of sub-systems defined by either functional or organisational parameters to support decision-making and control the organisational requirements. In the last two decades, businesses organisations around the world have employed configurable systems that attempt the integration of business processes and accommodate real-time data sharing (Vernadat, 2003). These systems are often known as Enterprise Resource Planning (ERP) systems and represent a solution that integrates business processes and data into a unique system to be shared across the various departments of the enterprise (Davenport 1998). According to Davenport, ERP definition emphasises the integration, between different division within the

organisation like production, procurement, sales and distribution, finance, and human resources.

The ERP application improves organisational performance and enhances competitive advantages among so many organisations (Davenport 1998). (Al-Mashari, Al-Mudimigh, and Zairi 2003) Argued that ERP systems have been recognised as one of the most common IT solutions in organisations, and the functionality they offer can serve a large variety of organisations. Lucas (Lucas Jr, Walton, and Ginzberg 1987) stresses that information technology such as ERP systems concept is utilised to capture, retrieve, transmit, manipulate, display or store information, by one application.

An ERP system can help business organisations in many ways. According to (Umble, Haft, and Umble 2003, Holland and Light 1999), (Sumner 2000) one of the most important features of ERP system is to automate the organisation's operations and supported the end-to-end business processes from best practices to maintain greater managerial control and fast decision-making. Another feature is that all information becomes centralised in a single relational database accessible to all departments, which eliminate the need for entries of the same data multiple times during the procedure (Muscatello and Chen 2008)

Typical ERP systems have been developed to include multi-modules in one application software such as sales and order management, marketing, purchasing that assists an organisation to execute its business functions (Tarn, Yen, and Beaumont 2002). These modules can interact with each other directly by updating a central database.

A business organisation has some choices to make in order to implement the best ERP system that matches their needs by selecting among various modules. According to (Rashid, Hossain, and Patrick 2002), ERP modules can work as stand-alone units, or can be combined to provide an integrated system. For example, they can choose and install only the modules they need from one or more ERP vendors, and they can combine their existing legacy systems and new ERP modules, or they can configure a system founded on a vendor’s special strengths. Sometimes, organisations can add or customise the functionality offered by an ERP vendor with additional modules or function from another ERP vendor (Light 2005).

2.2.1 ERP Systems Benefits

ERPs are becoming the largest and fastest growing systems in the software industry (Tarn, Yen, and Beaumont 2002 Al-Mashari, Al-Mudimigh, and Zairi 2003, Hillman Willis and Hillary Willis-Brown 2002). Year by year, it has been observed that the number of organisations using ERP systems is increasing around the world. The goal of ERP systems is to support the enterprise's operation at all levels and across all functions and processes. A well-planned and managed ERP implementation can increase the organisation's performance extensively across various departments. Markus & Tanis (Markus and Tanis 2000) summarized the benefits of ERP implementation, which describes the sources of the business benefits that ERP adaptation may bring.

Every large company has huge quantities of data, which are kept, in many repositories and, the information is distributed across many separate computer systems rather than just one system. Despite that, some organisations want to reduce redundancy and variation in data during transferring, recycling and reformatting the form of data from one system to another. Therefore, such organisations require implementation of ERP systems as a solution to support the integration of different business units through a central database of corporate information.

From the operational perspective, ERP systems give organisations an opportunity to increase sales and revenue; reduce high-cost structures; improve responsiveness to customers; face tough competition in the market; expand business globally; improve insufficient business performance; support new business strategies; simplify ineffective and complex business processes; and standardize business processes of an organisation (Davenport 1998) (Bingi, Sharma, and Godla 1999), (Sumner 2000), (Al-Mashari, Al-Mudimigh, and Zairi 2003).

Despite the benefits of having an ERP system, the implementation of an ERP project is lengthy and costly. The cost factor depends mainly on the size of the enterprise and the number of modules that organisations are willing to implement. ERP implementations can be either company-wide or limited to one major division. So a large number of ERP system applications have been implemented on a broad range of business organisations.

2.2.2 ERP Systems Architecture

According to (Shang and Seddon 2002), an ERP system pinpoints the importance of centralized management planning that utilizes resources effectively to achieve the communication between internal and external business. The ERP architecture aim for expediting the flow of information and communication between entire business functions and roles within the organisation and manage the relationships with external stakeholders (Rashid, Hossain, and Patrick 2002). This section introduces the system architecture that ERP systems are based upon. Understanding ERP system architecture and the functionality will provide a thorough inside of ERP technology and how it works within the enterprises.

Most current ERP system is designed based on Client/Server processing which the processed work will be shared between two computers Client and Server. The client is the presentation logic for users while Server is the processing and storage logic. In a Client/Server system, a number of client devices determined by end users such as desktop PCs request services from application servers, which in turn get the requested service-related information from the database servers. The requests may be data values, simple data files, service, communication, master file updates or transaction processing.

Three-tier ERP architecture was proposed as a solution (Manuel and AlGhamdi 2003) that supports the connection between databases, the business logics, and the end users interface. Thus, the three-tier ERP architecture consists of Database layer, Application layer, and Presentation layer (Graphical User Interface). The presentation layer is the most visible layer of an ERP software product while the application and database layers are typically developed and configured with ERP vendor specific standard (Selmeci et al. 2012).

Database Layer

In the database layer, numerous programs within the system process the data and the result are presented to the end user. This layer serves to manage and maintain the organisation's operational and transactional data include metadata where the business data and the entire repository are stored. ERP systems use a relational database to save data about their products, customer, employees, and vendors through various tables (Smets-Solanes and De Carvalho 2003). In ERP system, this data is referred to as master data. Once the ERP

system goes into production, every activity is saved to the database (i.e. the execution of the procedure for purchasing activity such as buying products from suppliers will create or updates one or more data in a database. This data is classified as transactional data (Chen 2001). In ERP system, master data changes slowly while the transactional that describes the business events changes rapidly.

Application Logic

The logic or processing layer is a second tier in such architecture. This layer represents controlling and process application business rules, function, logic and programs acting on data received/transferred from/to databases servers. This layer executes the instruction from the end user by transferring and receiving data from a database, validates the data and applies the business rules (Rashid, Hossain, and Patrick 2002). In application logic, a fully adopted ERP system involves in various business processes each of which poses a sequence of procedures within a functional department (Wang and Xu 2009). An end-to-end business process begins with its starting function, proceeds one by one function or sub-process, and ends with its finishing function (Johannesson and Perjons 2001).

As functions and associated business process are interrelated, the business transactions reference each other resulting in a significant complexity interrelationship. A typical process integration model of ERP consists of sales, production planning, quality assurance, payment, purchasing, outsourcing, and production process each of which in turn carries a set of functions. For example, the sales process starts with order entry function and finishes with invoicing function. It is associated with three other business processes such as production planning, quality assurance (QA), and payment process.

Presentation Layer

In ERP architecture, the presentation layer forms a third layer. It offers the end-user interface and provides a relevant physical layout of the application on a multitude of devices ranging from workstations to mobile device. This layer plays the front end or browser for data entry to access system functionality. As ERP end users interact with the presentation layer, data is transmitted, read, written, deleted or updated in the data layer. The presentation

layer here handles the execution of process and represents the functions data input/output, with a graphical user interface (Ollinger and Stahovich, 2004).

Having defined the EPR system architecture model and how it interacts with end users, then the next section looks into the implementation process of this system and addresses some issue related to the adaptation of the ERP system with business organisation requirements.

2.2.3 ERP Systems Implementation Lifecycle

Implementation of an ERP system is a complex process and, requires not just changing the technology but also various organisational adaptations to be able to make the best use of the system. To develop such a system requires planning, requirements analysis, design, detailed design, implementation and maintenance support in order to have successful ERP system (Somers and Nelson 2004).The implementation contains a procedure of configuring of the ERP package and adjusting it based on the business requirements of the organisation (Pajk, Indihar Štemberger, and Kovačić 2010). Therefore, the requirements analysis is a critical phase of the implementation life cycle, because of the significant investment in such a system for the organisation. In such a case the requirements analysis and the design phase play an important role to improve a successful implementation of an ERP system. At the design phase, the ERP developer determines the best practice, which the system supports. This integration pushes a company towards generic business processes; often refer to "best practices" by enterprise software vendors.

In many cases, the system will enable the company to operate more efficiently than it was before. Because this pushes organisations towards centralization and generic business processes, that the businesses must often modify their procedure to fit with the ERP system (Davenport 1998). This is known as re-engineering of the organisation's business process in order to implement the proper application. Re-engineering of business process activities focuses the organisation on identifying and improving the efficiency of critical procedures in the process, removing inefficient operations and restructuring necessary procedures into standard form (Muscatello and Chen 2008).

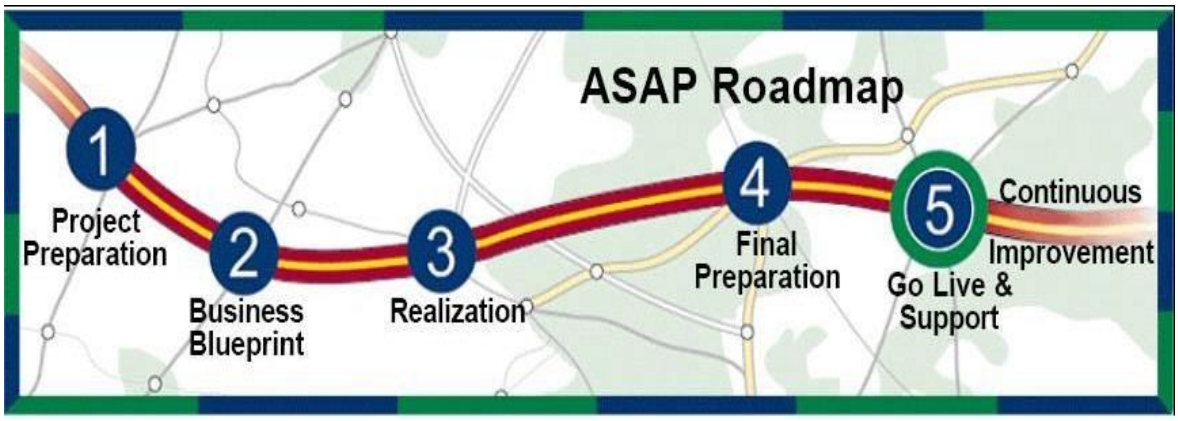


Figure 2-1 ASAP Roadmap (Gulledge and Simon 2005, Miller 1998)

Implementing a suitable ERP system is an important development requires a significant level of resources, commitment, and adjustments across the organisation and has certain basic phases (Umble, Haft, and Umble 2003). The following section describes an example of an implementation methodology that is based on a process model used for implementing SAP ERP systems. According to ASAP (Gulledge and Simon 2005, Miller 1998) described the implementation roadmap, in the following phases (see Figure 2-1):

1. Project preparation: the planning stage for the ERP project, where crucial strategic decisions about project goals, implementation scope, schedule, budget, implementation sequence need to be made, and the project organisation and relevant committees established

2. Business Blueprint: This phase in ERP implementation refers to documentation of the company's requirements, and how the organisational structure, business function, and processes is expressed in the ERP system.

3. Realisation: This phase in ERP implementation configures the requirements contained in the blueprint into the system. The configuration of the scope can be arranged for up to four cycles, starting with the major scope and reaching further levels of detail in later cycles. Integration testing and end user documentation are the key activities in this phase.

4. Final preparation: This phase is completing the preparation work, testing, end user training, system management and remove unnecessary activities. All issues (i.e. such as gaps in business requirements, configuration, user resistance, end user training and support) must be resolved, and the requirements for the go-live need to be fulfilled, before proceeding to the next phase.

5. Go live and support: denotes the move from a configured environment to the system operation. The most critical activities are setting up system support function, monitoring the system transactions, and optimising overall system performance.

As (Hillman Willis and Hillary Willis-Brown 2002, Fui-Hoon Nah, Lee-Shang Lau, and Kuang 2001, Davenport 1998) argued against the prevailing assumption of treating ERP as a project that has a termination date. Besides, the successful implementation is not the end of the ERP journey, as the post-implementation phase is where the real challenges begin. The post-implementation phase includes critical processes such as testing the system for effectiveness (i.e. it's actual, versus projected, or checking the compatibility with business processes), data integrity, checking the reliability, system utilization and most importantly, assessing and evaluating the benefits of implementation of the system (Holland and Light 1999) (Fui-Hoon Nah, Lee-Shang Lau, and Kuang 2001).

During post-implementation phase organisations often face a broad range of issues like technical pitfalls, unexpected business requirements, inadequate user's behaviour and deficient system design. In order to address such challenge ERP system requires some adjustment to overcome this problem (Daneva and Wieringa 2008). Compared to the research in ERP implementation, the study in ERP post-implementation has long been considered under investigated but not given a proper recognition especially when the organisations face with the inevitable change in requirements that lead the modification of the ERP system. The following section argues this issue in further detail and attempts to define and classify the alternative solutions on how to overcome this problem during the post-implementation phase.

2.3 ERP Post-implementation

To date, change in requirement at ERP post-implementation has received relatively little attention in the literature (Law, Chen, and Wu 2010), (Grabski, Leech, and Schmidt 2011) (Zhu et al. 2010, Ifinedo et al. 2010), (Yu 2005). Besides, the literature tends to focus on CSF (Critical Success Factors) of the post-implementation phase and, in particular, how these differ from CSFs of the implementation phase (Ram, Corkindale, and Wu 2013), (Ram, Corkindale, and Wu 2013), (Moalagh and Ravasan 2013). The quality of the implemented

system and the data it uses are frequently identified as technical CSF of ERP Post-implementation (Zhu et al. 2010, Ifinedo et al. 2010). Poorly managed to post-implementation modifications reduce the ERP system and data quality, leading to chaotic systems that are difficult to control (Zhu et al. 2010, Ifinedo et al. 2010). Therefore, there is a need to support stakeholders, such as business analysts and developers, to assess the impact of ERP post-implementation changes to guide them during the implementation of the proposed modifications.

Like traditional software system, ERP systems require maintenance (Ng, Gable, and Chan 2002). Post-implementation modification concern changes to align the ERP system to the business requirements (Ha and Ahn 2014). These changes usually concern the static design structure of an ERP system. The ERP Post-implementation activity represents the longest and the most expensive phase in an ERP system lifecycle (Lübke and Gómez 2010). Despite the importance of ERP post-implementation being recognised by prior studies, there has been little research exploring on how to control and access the modification of an ERP system from a design point of view other than the management aspect. Thus it is important to know this issue and consider how they may influence the ERP post-implementation success. As such, the rest of this section attempts to answer these two questions:

- What are the reasons for ERP system modification after post implementation?
- And, how does the post implementation modification apply to the ERP system?

2.3.1 Reason for ERP Systems Change

ERP systems face with inevitable changes during post-implementation phase to align the ERP functionality to the business requirements (Oseni et al. 2014, Themistocleous et al. 2001). Because ERP systems consist of functionality intended to meet the needs of a wide variety of customers (Kumar and van Hillegersberg 2000), there are always gaps between the delivered functionality of the product and the current business practice of the implementing organisation (Orlikowski 2002) (Dong 2000). Gap or misfit in the ERP system and organisations requirements is fundamentally caused by the initial design of the ERP system which supports the standard procedures in a certain field. While each organisation has its unique way of working, which is formed as a combination of company-specific, sector

specific and country/culture specific factor (Soh, Kien, and Tay-Yap 2000) which makes it different from ERP system procedures. Understanding of what creates misfit and how to explore the misfit issue throughout the life cycle of an ERP system remains unclear. Four types of misfits identified in the literature by (Hong and Kim 2002, Soh, Kien, and Tay-Yap 2000) as follows:

- Data misfit refers to incompatibilities between organisational requirements and ERP package in terms of the data format and the relationships in the data model.
- Functional misfits can be divided into access, control, and operational misfit. Access misfits occur when the necessary access to perform certain tasks is not present, control misfits refer to missing validation procedures or checking routines and operational misfits refer to missing operational steps or the presence of certain steps, which are inappropriate.
- Output misfit refers to incompatibilities between requirements and ERP package in terms of the information content and presentation format of the output
- Interface misfit occurs when there is a gap between the way the graphical user interface (look and feel) of the ERP package is designed and what the users are expected to work with.

Misfits in requirements that cannot be determined by configuration are often resolved in one of two ways: re-engineering of organisation procedures or ERP system modification (Brehm, Heinzl, and Markus 2001). As for the first alternative, change in procedure requires significant organisational change, resulting in high upfront costs (O'Brien and Marakas 2006). Therefore, customizations/ERP system modification may be initially seen as the most efficient solution. However, ERP system providers discourage a modification to any portion of their system and typically not supporting anything that has been customised (Brehm, Heinzl, and Markus 2001). Many studies advocate that the change in ERP systems should be implemented with minimal modification to the application (Somers and Nelson 2004); (Upadhyay, Jahanyan, and Dan 2011) as the modification of ERP system is problematic and may increase costs and limit maintainability (Alawattage et al. 2007). Each time a vendor patch or upgrade needs to be applied to an ERP, all customizations must be reviewed, reapplied and retested in the system (Yakovlev 2002).

While the misfit between ERP packages and organisational need is widely addressed in the literature (Light 2005), (Luo and Strong 2004) the implication for resolving the misfit by customising the ERP packages and its impact on the efficiency of such package have not yet been investigated. (Light 2005) introduces the top problem from various organisations that result in the customization and maintenance of the ERP system. (Fui-Hoon Nah, Lee-Shang Lau, and Kuang 2001) Identified maintenance activities concerning at ERP post-implementation phase. The activities include adaptive maintenance (transfer, testing, modification and enhancement) corrective maintenance (troubleshooting, and updating data model), perfective maintenance (version upgrades). As described missing or incorrect functionality, a complexity of using the system (GUI issue), structural and procedural problems are the most recognisable issue within the ERP project.

Despite the complexity of ERP system during implementation and post-implementation, the problem of flexibility in ERP system is recognised as the biggest challenges in ERP adoption and becomes more important than before to address the above problems (Wei, Wang, and Ju 2005). According to (Arteta and Giachetti 2004) that defines the flexibility in any system as the "ability to manage and apply knowledge effectively, so that an organisation has the potential to thrive in a continuously changing and unexpected business environment". Flexibility implies not only the ability respond to unexpected modification but also to act proactively with respect to applying with the minimum cost and effort (Arteta and Giachetti 2004).

2.3.2 Solutions for Change in Requirements

Many organisation experience various challenges when going live with ERP application, (Markus, Petrie, and Axline 2003). A research work by (Keil and Tiwana 2005) indicates; that the functionality of ERP system is the most important features of the standard business application to predict perceived usefulness of the system. Given the scope of ERP systems, the insight about the issue of the misfit from (Lucas Jr, Walton, and Ginzberg 1987) that "Either the organisation has to change its procedures, compromise on processing needs satisfied, or modify the ERP package" applies to ERP systems directly. Others (Soh, Kien, and Tay-Yap 2000) address this issue at the operational level that developed a framework for

misfit in requirement during ERP implementation (Soh, Kien, and Tay-Yap 2000). (Somers and Nelson 2003) suggests a number of integration mechanisms ensure the fit between ERP system and organisation needs as; project organisation, a business driven implementation, organisational and package adoption.

In the literature, various papers such as (Salmeron and Lopez 2012) propose models to support ERP modification and enhance the flexibility issue such as service-oriented architecture (Lechesa, Seymour, and Schuler 2012) business process management techniques (Kilpinen 2008) configurable process model (La Rosa et al. 2011) (Gottschalk, van der Aalst, and Jansen-Vullers 2007) or by extending ERP system with a third application to support the organisation requirement and apply change in ERP system. Another alternative solution such as modification of the ERP system through customization of ERP source code is suggested by (Luo and Strong 2004) however these approaches brought the highest risk and cost for most of the organisation.

The ERP packaged modification could potentially meet the user requirements. Numerous research and industrial studies of the critical success factors for ERP implementation success perceived that the preferable approach to implementing ERP systems without modification of the ERP package (Nah, Zuckweiler, and Lee-Shang Lau 2003). However, sometimes due to strategic alignment, and competitive market in the business environment, modifications of an ERP system is necessary. According to (Scott and Kaindl 2000) outline that 20% of the processes in an organisation cannot be represented in the ERP systems without performing customization. The modifications are essential for the ERP system to meet the requirements of the organisation; however, the issues associated with customization are far reaching to adjust the requirements.

2.3.3 ERP Package Modification

The customization/modification ERP system classifies into two models known as internal change and external change. The internal modification refers to the change in the system such as configuration and code modification whereas the external modification relates to the improvement of the system by using an external entity or adaptation of another system without changing the current ERP system. (Glass 1998) Categorised ERP adaptation

types into ERP customization, extension, and configuration. ERP customization refers to the modification where the organisation requests to change the system through code modification. Extension refers to the modification where the organisation asks from a third party vendor to use some particular functionality in parallel when they are running the ERP system.

Lastly, the configuration is the type of modification that only requires adjusting some parameters setting without changing the existing system. Follow description starts from the most expensive and high risk to the least and easier form of modification perspective. ERP system customization is appropriate for those companies that believe their business processes are better than those implemented in an ERP system and do not want to lose their competitive advantage (Štemberger and Kovačič 2008) therefore they select from various modification strategies in case that the solution could not support the ERP system.

There are so many tailoring solutions available to resolve the potential gap between ERP and organisational requirements studied by (Brehm, Heinzl, and Markus 2001). The business analyst must make a choice among these solutions. However, empirical insights into how these common tailoring options or strategies relate to ERP maintenance efforts remain unavailable. In Chapter 6 of this thesis describe a preliminary effort to quantify the relative importance and effort of different tailoring strategies through the study with ERP expert.

2.3.3.1 ERP Modification Strategies

Table 2-1 provides a summary of ERP tailoring strategies following an example (Brehm, Heinzl, and Markus 2001). In addition, in the next sections, we explain the detail description for each customization strategies in order to analyse these options in more detail.

Configuration

Configuration is the tailoring type with the lowest possible impact values. ERP configuration involves selecting from the reference model and setting the parameters that allow the organisation to adjust and modify the system within the boundaries, to reflect requirements without changing the ERP source code (Brehm, Heinzl, and Markus 2001). Parameterizations play a significant role in the customization of a reference model. It enables

the parameters or variants of a reference model's features (such as processes, functions, data) to be set according to an enterprise's requirements (specific business processes and policies) (Leyh 2011).

Table 2-1 ERP Customization Option

Type of modification strategies	Description	Example
Configuration	Modifying the setting of parameters of the ERP system;	Define organisational units; create standard reports; formulate available-to-promise logic; use of a standard interface to an archive system
Bolt-on	Extended and packaged functionality developed to function with the ERP system created by a third party supplier.	
Workflow Programming	Allows modifying the ERP standard workflows;	
User Exits	Places in the ERP package code where tailor-made code can be placed to extend the functionality	Develop a statistical function for calculating particular metrics
ERP Programming	Adding applications developed in the ERP system programing language without changing existing code of the EPR system	Create a program that calculates the phases of the moon for use in production scheduling
Interface development	Implementing the change using external functionality that is accessed through the implementation of standard interfaces	Interface with custom-build shop-floor-system or with a CRM package
Package Code Modification	Implementation of change, whereby the ERP package source code is modified ad-hoc	Change error message in warning; Modifying production planning operation

The various options are run through the system parameters, whose values are determined during the enterprise system implementation. The configuration of an ERP system requires some consulting work, but without custom code development. If a gap can be filled through configuration, the costs and risks are much minimised (Yen, Idrus, and Yusof 2011). However, sometimes this option requires more time and effort for configuring complex cases or business process in the system. In this situation, it is better to acknowledge understanding of the impact of additional customization strategies.

According to (Uppstrom et al. 2015), recently with the help of technology, configuration enables changes not only by setting parameters but also changes in the User Interface or databases, through drag and drop. The example of parameterization is a Boolean

parameter indicating either the functionality performs in the system or not by setting the value to true or false. Such that by enabling warehouse location control functionality all the warehouse location related processes, like warehouse locations management and receiving purchased goods to locations, are also enabled (Soffer, Golany, and Dori 2003)

Bolt-on

The placement, of third party solutions, also called bolt-on or add-on solutions (Pajk and Kovacic 2013). If the ERP system could not meet the business need, it is possible to deploy a third-party vendor known as an independent software vendor solution. Although the different vendor can provide a vertical or industry solution, that supports organisations particular needs. Typically, some functions are not included in the scope of the ERP system, but this can be enhanced through other business information systems like Product Lifecycle Management Systems, Data Warehouses, and Advanced Planning Systems.

These systems are typically interacting with the ERP system through business processes and interfaces (Munkelt 2013). In order to guarantee the successful execution of business processes across system boundaries, the collaboration and communication layer of these systems with the ERP system has to be carefully monitored, and maintain through some interfaces.

Alternatively, organisations can implement specific third party packages that are designed to work with the ERP package and extend functionality (Rothenberger and Srite 2009). However, implementation and upgrades of these systems are managed separately (Munkelt 2013). The third-party solution introduces new complexity in order to integrate with the ERP system. Despite that these solutions may not be compatible with the ERP system, in that case, this requires some adjustment to the ERP systems. Besides for solving this problem, the misalignment between the ERP system and bolt-on version can also be an issue when upgrading the ERP system to a new version (Brehm, Heinzl, and Markus 2001).

According to (Hsu, Sylvestre, and Sayed 2006), if bolt-on is designed by the same ERP vendor or built for specific ERP systems, the flexibility could be higher due to it is based on an external fragment of the ERP system. However, if it is designed by another vendor(s) to work together with any ERP system, then the compatibility problems may cause

more risks because ERP system needs to develop interfaces to communicate with bolt-on solution. For example, an ERP system is selected from SAP vendor for one business organisation and runs a couple of modules in their systems. The business organisations also believe that the financial accounting module of a third party ERP vendor is more match with their requirement than the existing ERP system. Despite that, the SAP is not supporting the add-on of third party ERP vendor. Therefore, this requires of developing a new interface with the existing ERP system through ABAP Programming so that the business organisation can take advantage of using third party module.

Workflow Programming

Workflows in the ERP system are used to model business processes, e.g. define steps for handling of procurement and data that produced. Workflow programming will be required if the ERP standard workflow is not sufficient to comply with the adopter's needs. If a new function or process is programmed, or the implementation of the existing process is changed the workflow programming will be involved, and the impact of tailoring categories will be possibly higher compared above. According to (Luo and Strong 2004) typology sometimes to write industry workflows, may also require source code modification that known as a separate category.

Some other example for workflow programming could be:

- Mapping workflow states to tasks in the enterprise
- Changing the rules, that control if a transition gets activated or
- Adding intermediate workflow states to support more complex decision processes of an adopting organisation

User Exits

In computer software, a user exit is a form of software program where organisations can arrange for their tailor-made program to be utilised. This is limited to some range of functions, process, documents that have been pre-defined by the ERP vendor to be able to adopt the new requirements. Typically, not all customization requirements can be satisfied through parameter setting. In many cases, in particular, when the extension of ERP system

standard functionality is involved, company specific programs suggest being written and embedded (Kurbel 2013). For this reason, ERP provides a large number of pre-defined user exits. The exit allows the organisation to extend a system with their requirements such as master data, functionality or a different method of calculation which all defines through an open interface.

Sometimes when a modification in requirements is needed to provide specific functionality, through the development of add-on modules that are plugged into the ERP's user exits. Besides, subsequent versions of the ERP software may not retain the same user exits during the upgrade (Soh, Kien, and Tay-Yap 2000).

ERP Programming

To address different customization needs, organisations can also develop custom features on top of their ERP platforms by applying through the ERP system language or standard programming languages (Rothenberger and Srite 2009). This type of customization is also known as enhancements refer to improve the functionality of the ERP package through the language provided by the vendor, although it does not require modification of existing system code. Both user exits and ERP programming a new function with a different document type can be designed in ERP in a way that is closed to configuration than to programming.

Interface Development

Interface development does not mean to change the graphical user interface, layout or re-positioning of interfaces of the ERP system. This type of customization evolves with communication or bridging the gaps between two different systems to become consistent such as the third party system.

The term integration implies on all relevant data for a particular bounded and closed set of business processes from a third-party or external system that is performed with ERP by enhancing the system through interface development (Gulledge 2006). The integration of external system with the ERP system could classify into following forms:

- The Point-to-point integration is the expensive one due to developing pairwise interface among system.
- Database-to-database integration that only requires sharing data at the database level. Although enterprise applications publish application program interface that allow interfacing at the application level that prevent integrity problem when updating a database.
- Data Warehouse integration is used to mapping data from any databases with different models or schema.
- Enterprise application integration is sharing business Process logic and data across via solution offered by third-party vendor that connect multiple systems at the application or database level.

Interface development is an expensive solution and requires a significant portion of the cost for any enterprise system implementation. In addition, there are a few issues concerns as the complexity and managing the scope of integrity through multiple data source (Gulledge 2006). Furthermore, any modification or upgrade to one system can be expected to result in complex and costly assessment by re-testing, redesigning or event coding of the interface.

Code Modification

Unlike Vanilla ERP implementation that does not have any customization option, offers modification option which means the organisation can adopt the ERP system to specified requirements without modifying their legacy system. ERP system source code can be changed to fit an organisation's needs and requires substantial development effort and specialised expertise (Rothenberger and Srite 2009). This type of customization is problematic since the code modification may need to be re-implemented and tested when the system is migrated to an upgrade version. Package code customization should theoretically not happen in an ERP implementation process. ERP design philosophy is intended to make the package code general enough to accomplish adaption to different business needs by employing lighter tailoring types than code modification. Although this form of customization is not recommended by ERP vendors and consultant in the end completely

satisfies the organisation's requirement. This is because of every time that the ERP vendor releases a new version of the ERP system, and these modifications also require to apply to the upgraded version as well which again needs substantial development effort and cost for the organisation. The following sections argue this issue in further detail.

2.4 ERP Systems Modification and Change Management

There is a fair amount of recent research regarding the challenges of ERP implementations (Akkermans and van Helden 2002), (Akkermans et al. 2003), (Sun, Yazdani, and Overend 2005), (Ifinedo and Nahar 2007), (Palomino Murcia and Whitley 2007), (Soja 2009) however research addressing the issues associated with the requirement modification after ERP implementation is lacking (Ifinedo and Nahar 2006). Furthermore, ERP modification poses a significant risk to the project success (Parr and Shanks 2000); (Finney and Corbett 2007) and threatens the return on investment by increased implementation and maintenance costs (Beatty and Williams 2006). Many organisations still lack experience and expertise in this area, and there are no proper guidelines or standards for ERP modification and upgrade preparation or step-by-step procedure to assist practitioners. Uncontrolled or poorly managed ERP post-implementation changes may lead to low quality, chaotic systems and data that are difficult to use and maintain. These are likely to result in a bad decision making and business process performance, which ultimately minimise an organisation’s profitability and productivity (Ifinedo et al. 2010, Yu 2005) we first need to understand the change management process and its activities. The next sections introduce a brief description of change management terminology and change management processes reported in the literature.

2.5 Change Management Terminology

Phrase change management is very common in the management field. Change management, one of the earliest approaches by (Harked, Eason, and Dobson 1993) classifies requirements into stable/permanent requirements that originate from the core activity of the organisation relate directly to the domain of the system (e.g. flying airlines or treating patients) and the other is changing/unpredictable requirements as the requirements that are likely to change during the system development process or after the system become

operational. This type of requirement is also known as emergent requirements, consequential requirements, adaptive requirement, and on-demand requirement. For instance, in the ERP post-implementation phase, a change in requirement may be driven by internal reason, e.g., a new and more efficient warehouse management policy suggested by the management team that must be reflected in the ERP system. The cause of change in requirements can be classified as the correction of error, the improvement of components or addition of functionality to the software system. It is important to identify the type of change as it helps to identify different approached in order to deal with them.

According to (Finney and Corbett 2007) (Umble, Haft, and Umble 2003), the business organisation should be adequately prepared for unexpected change with proper change management techniques. Change management is the process by which the business organisation can accommodate their new requirements. Change management encompasses the effective strategies and program. Huang (Huang, Yee, and Mak 2001) focused their research on the management of change processes and the solution for improvement of software tools to support modification.

Change management has two objectives the first is to provide support for processing of changes and the second is to track and monitor all active and implemented modification. As the first objective, the process involves the identification of change, assessment and analysis of the impact, prioritisation, and planning for change implementation, and finally decision for the rejection or postponement of the requested modification. The second objective is more about the traceability of change. Companies approach change management quite differently due to the specific requirements, but they have a similar process (Pikosz and Malmqvist 1998). The terminology may differ from enterprise to enterprise, but the purpose of the stages is similar. The next section explains the process of change management that applies to the most of the change management terminologies.

2.5.1 Change Management Process

Over the years, several software system modification framework and models have been proposed, with common activities in order to apply the change into their system. Formal processes and routines should govern these change management process so that

modifications to the configured system are handled in a consistent. The change management process discussed here is first taken from (Leon 2000) (Jönsson 2007) that involve the following phases:

- Change initiation and determination and
- Change classification,
- Change analysis,
- Change acceptance or rejection, and
- Change implementation and verification.

In Change process model, a modification of the requirement is usually identified by a change request. When a change is initiated, a change request is created to track the change until it is resolved and closed (Crnkovic, Asklund, and Dahlqvist 2003). The change request is seen as the initiating entity, which means that one change request can prescribe changes to many items. Thus it is essential to classify the type of change based on the taxonomy that allows different change to be dealt with and analysis differently.

Throughout the literature, authors have been using slightly different terms for modifications or changes to a software product. Terms that have been used include engineering design change (Leech and Turner 1985) product change (Inness 1994) design change (Guenov and Barker 2005) product design change (Huang and Mak 1998) redesign (Ollinger and Stahovich 2001) engineering change (Jarratt et al. 2011). Although these terms refer to the same phenomenon, often have different interpretations.

Despite to what mention before, the change management is also important in the field of manufacturing and particularly in product lifecycle management that is confronted with many changes. Engineering Change Management as an approach by (Terwiesch and Loch 1999) which draws a clear line between design iterations and engineering change by restricting the latter to the post-implementation phase. Engineering change management deals with "the organisation and control of the process of making alterations to products" (Jarratt 2004). It should not be confused with 'change management' that is common in business and management literature. (Wright 1997) Restricts the meaning of an engineering change to "a modification to a component of a product after that product has entered production". This follows a common conception that engineering changes and their

associated processes occur after the design has been completed and hence the production has been started.

2.5.2 Engineering Change Management

The discipline of engineering change management has gained increased popularity within system engineering (Hamraz, Caldwell, and Clarkson 2012). Early research on engineering changes was focused mainly on improvements in project management techniques and optimisation of design processes (Wright 1997). The author (Wright 1997, Jarratt 2004) published an extensive review of this early research into engineering change management. He also proposed a six-phased engineering change request process (ECR) (see Figure 2-2): (i) identification of change request, (ii) determination of possible solution(s), (iii) impact assessment of ECR solution(s), approval of a solution by change committees, (v) implementation of solution, and (vi) review the particular change element.

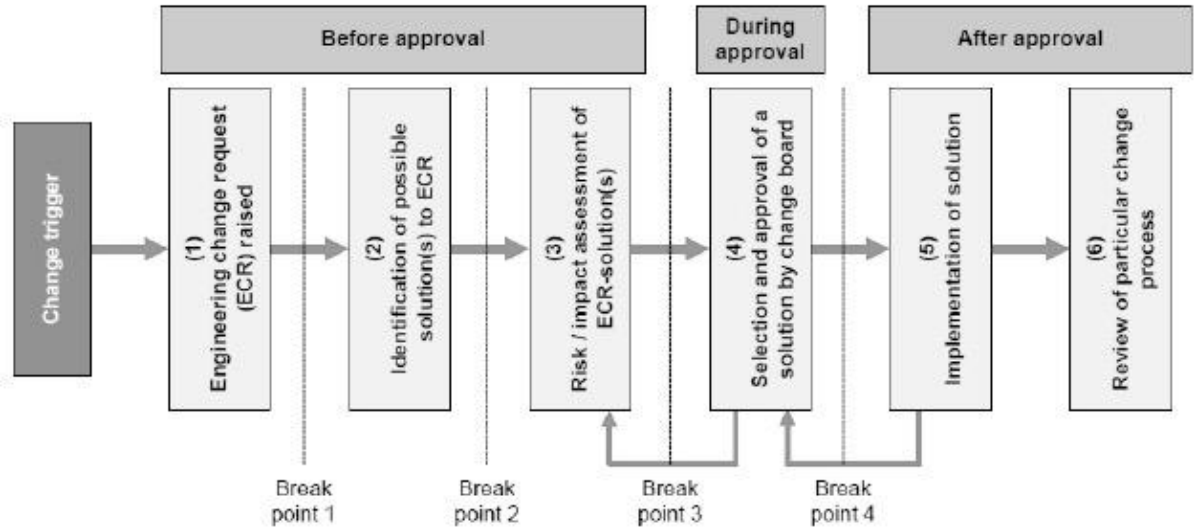


Figure 2-2- Engineering change process (Jarratt 2004)

The example of Engineering change process begins with establishing the user requirements (Eckert, Clarkson, and Zanker 2004). This follows a process where experts will make the major design decisions. Subsequently, a proposal is presented to the analysts who assess the requirement and priorities the modification task. This also talks about the beginning of change processes for the emergent changes. In the next stage, the analyst and

the change engineers investigate the impact of the change in detail will raise a formal engineering change request. Then, individuals or teams are chosen for each system involved that will further assess the change and will propose solutions to their cost and implementation planning. The solutions are later discussed in a joint meeting of all involved teams and individual where the preferred solution is agreed. Each of the involved teams can then continue with the implementation of their part of the change.

Change management process in software system topic has gained considerable attention to amend the modification. In recent years, the focus of research has shifted from methods to manage the change process to methods and models that aim to predict the effort estimate of applying the change. Effort estimate is crucial activity in managing requirements change, particularly in analysing the change impacts. In the change process (Bohner 1996) (Bohner 2002) (see Figure 2-3) describes the Impact Analysis as a critical phase of new product development and software maintenance.

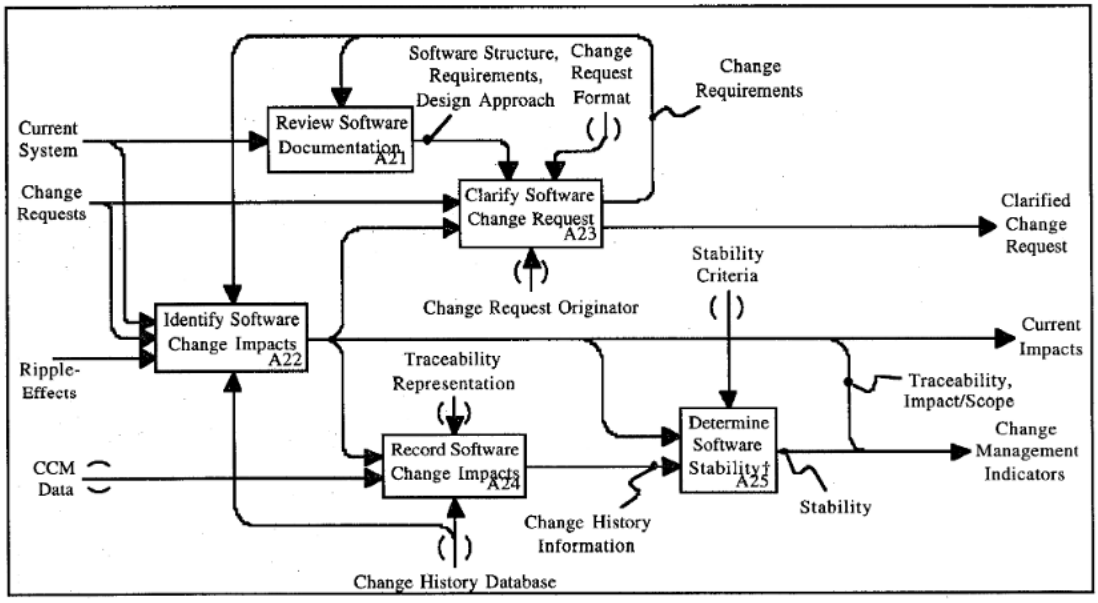


Figure 2-3 Understanding Software Change and determine Impact (Bohner, 2000)

Given that Impact Analysis (IA) can be applied in these different cases, both of these contexts are considered in this research. (Bohner 1996) also, indicates that designers perform IA in response to a change request (also known as an engineering change or engineering

change request) to scope the modification that is approved by management or change control board.

2.6 Impact Analysis Approach

The motivation behind the impact analysis process is to capture the software components that are likely to be affected by the modification (Kama 2013). According to (Rajlich and Gosavi 2004) running impact analysis is a significant step during modification and maintenance of software application. It enables to predict the amount of effort that is required in order to implement a change in a software system (Bohner 1996).

(Bohner 1996) studied the foundations of impact analysis, and contributed the following definition which has been cited by most researchers in this field:

"Identifying the potential consequences of a change, or estimating what needs to be modified to accomplish a change". (Bohner 1996) Pg. 3

Impact analysis is recognised as the lifecycle of any software system that is accompanied by frequent changes, which are commonly referred to, as Software Evolution (Lehnert 2011b). During impact analysis, in two main questions needed to ask to perform the modification.

- First which element in the software system need be changed?
- Second which other elements are impacted by this modification?

According to (Queille et al. 1994), the goal of impact analysis is to minimise the unexpected side effect of change. Considering the work by (Bohner 1996), further research has offered and explored a wide range of techniques and methods for change impact analysis and extended this area from program code to other types of software artefacts such as requirements (Von Knethen 2002), design model (Bengtsson et al. 2004), or test cases (Kung et al. 1994). More extensively, the input of impact analysis is the change set (i.e. can be either at the design specification level or source code) while the output is the affected item set (Sun et al. 2010). (Lehnert 2011b) reviewed impact analysis research and classified into three fields based on the criteria of the taxonomy of modification as source code, architecture or design model, and miscellaneous artefacts.

The impact analysis approach at the source code level can be either performed in the form of static or dynamic (Rohatgi, Hamou-Lhadj, and Rilling 2008). Static analysis is practised and suitable for defining structural properties and performs at the design level to evaluate the consequence of modification in an abstracted model of the software system. Dynamic analysis is described in the behavioural properties such as component interactions that analysis can test the correct run-time behaviour of the component in the system (Ernst 2003). In particular, static analysis is limited to defining all possible executions, whereas dynamic analysis can involve the execution phenomena of a particular activity.

Recently other approaches such as model-driven techniques offer change impact on a representation of the software system at a higher level of abstraction than the source code (Lehnert and Riebisch 2012, Amjad Alam, Binti Ahmad, and Akhtar 2014). These models are further divided into architectural (Zhao et al. 2002) and requirements models (Yan, Li, and Sun 2012), to represent the earlier stage of software development, i.e. requirements capturing and structural reasoning. Other approaches can be used to analyse the range of documents and data sources, such as configuration files, bug trackers and documentation.

From the classification by (Lehnert 2011b, a) exposed that 65% of all studied literature is conducted with analysing source code changes and the impact of the modification of code. Only 11% analyse modifications and their implications on software architectures, and another 7% investigate the change on requirements. However, the current impact analysis techniques require source code or architectural representations so that it can predict changes. Despite that, such information is not available for all software systems or for the user who runs the impact analysis (e.g. a requirements engineer may find it difficult to analyse source code.) Therefore, it should be an approach to predict the impact so that the information is understandable to all domain users.

Research from (Aryani et al. 2009) introduces a method to determine the propagation of software change based on the information that is understandable and visible to domain users. The method runs without demanding access to development histories or source code, as it receives information from user manuals, and expert experience and stores them in a weighted dependency graph to facilitate the purpose of the change propagation.

The work of (Briand 2007) presents an impact analysis approach for architectural models, which drive from UML models and support the specification of the model. In this method, (Briand 2007) discussed the effect of modifications to indirectly related items and provided the estimate through a distance measurement. The approach applies to either remove the change propagation or to impact weight paths according to the depth of the modification.

Understanding software architecture model could facilitate impact of change analysis for two main purposes according to (Lehnert 2011a) first it allows to extract information from the architecture model to see where the modifications will appear, and second to determine the impact on the other parts of the model. For an enterprise system, however, this means that an architectural description language (de Boer et al. 2005) or BPMN (Business Process Modelling Notation), has to offer an integrated view of the entire application.

2.6.1 Impact analysis of Enterprise System

The main characteristic of impact analysis is to define entities, which are either directly, or indirectly affected by the change. Impact analysis has been considered a useful tool for planning changes, making changes, and tracing through the effects of changes (Rohatgi, Hamou-Lhadj, and Rilling 2009). The use of impact analysis could be implemented before or after the change implementation phase.

Change impact analysis can be executed by a single software system, but also on an enterprise level for full application landscape (Langermeier, Saad, and Bauer 2014). Impact analysis for an enterprise system is involved in assessing the impact of a change in any part of an enterprise across the business organisation. For example, modification of an enterprise's strategy can have multiple significant results in all layers of an enterprise model including business processes, data management, organisation structure and technological infrastructure (Sunkle, Kulkarni, and Roychoudhury 2013).

A simple modification in the system can potentially influence many different layers and domain of enterprise architectures. Enterprise architectures incorporate business processes, organisational structures, and the actual software architecture that are likely to be changed. (De Boer, Harink, and Heijboer 2002, de Boer et al. 2005) Analyse and evaluate

propagation of change on the enterprise level, and introduced an analytic method by conducting an enterprise architecture modelling language as ArchiMate. The method involves the business modelling language and UML concepts at a very high level and maintains a business layer, an application layer, and a technology layer. Design entities and dependencies for each layer are stored as a dependency graph to allow the execution of impact analysis, which is based on structural layers carried by ArchiMate.

Understanding and assessing the static structure of the components of software architectures is an essential task during impact analysis. In addition to that (Feng and Maletic 2006) proposed an approach to maintain both static and dynamic impact analysis at the architectural level. Their approach determines the architecture elements causing the change and affected by the change. Feng and maletic are first introduced the taxonomy of modification and then define a set of impact rules to capture the impact of the particular modification. Moreover, this approach explains interaction item tracks from component and sequence diagrams, which are divided by impact rules to obtain the set of affected items.

2.6.2 Impact Analysis of Business Processes

In enterprise system business processes experience a lot of modification; that must be reflected in the source code of the current application. Representing such a modification to the source code is not small. (Xiao, Quo, and Zou 2007) studied impact analysis in the area of business processes in the service-oriented applications. Their approach involves the analysis of requirements encoded in BPEL, and source code via dependency graphs. Several methods and paradigms by (Biswas et al. 2011) (Weske 1998), (Casati et al. 1998), (Zhao and Liu 2013), (Reichert and Dadam 1998), (Weber, Sadiq, and Reichert 2009), (Kherbouche et al. 2013), (Mafazi et al. 2013) deal with the evolution of the business process of enterprise application and propose different strategies during the execution of the system on how to treat process instance when there is migration. (Weske 1998) offers a flexible Modelling for workflow activities based on a meta-model of business that handles dynamic modifications. Another approach proposed (Zhao and Liu 2013) suggests version management for business process schema evolution by explaining different business process schema evolutions and the dependencies. Furthermore, the work ADEPT-flex by (Reichert

and Dadam 1998) represent a graph based workflow model to support the integration of dynamic changes. In the same way, (Sun and Jiang 2009) to calculate the minimum region affected by the modification.

Even small modifications can negatively affect other parts of the system, based on (Brehm, Heinzl, and Markus 2001) the effect of customizations is often not realized until significant resource consumption has already occurred. Each time an ERP organisation patch or upgrade needs all modifications and changes must be reviewed, reapplied and retested (Yakovlev 2002). Impact analysis is an important task in the ERP post-implementation and evolution. It identifies the set of entities that need to be modified and transformed to enhance an existing system with a new feature.

One of the most important kinds of analysis of an enterprise is the assessment of the impact of changes (de Boer et al. 2005). (Every single part of an enterprise is subject to change, and each change may have significant consequences within all domains of the enterprise. A lot of effort is therefore devoted to maintaining the integrity of an architectural description.

A recent software system like ERP is developed in a more complex way by incorporating with more features and newer technology that the needs for emerging impact analysis become an important issue. The approach helps the ERP system to be able to identify the scope of the change and the complexity of the change.

2.7 Summary

This chapter addresses the concepts used throughout the thesis and explores the state of the art for the current research. As mentioned before, ERP system is a large-scale application that integrates business processes and data of the business organisations. Once implemented is completed, the organisation faced with inevitable change during the post-implementation phase to align the ERP functionality based on business requirements. To date, ERP post-implementation changes have received relatively little attention in the literature. The literature particularly tends to focus on CSF (Critical Success Factors) of the post-implementation phase and, how these differ from CSFs of the implementation phase. Uncontrolled or poorly managed ERP post-implementation changes may lead to low quality,

chaotic systems, and data that is hard to use and maintain in future. However, the literature lacks constructivist approaches to change management to support ERP stakeholders during the post-implementation phase. This thesis argues for the needs to provide ERP stakeholders, such as users, business analysts, and developers, with scientifically grounded principles, methods and tools to facilitate the assessment, and implementation of the ERP changes.

The software systems change impact analysis as a discipline in software engineering focusing on understanding, predicting and possibly quantifying the impact of source code modifications (Bohner 2002), (Hassan and Holt 2004). Various techniques for analysing the impact have been discussed in the literature, while these techniques provide an excellent example of how to apply the impact analysis in specific domains, it can be hard to implement them to assess the modification of the ERP system.

For instance, a change in the process of purchase order may affect all the open order cases with suppliers. Hence, the impact of this change on the ERP system should be carefully analysed. Furthermore, the associated running instances affected by a change should be allowed in such a way to terminate safely before an implementation of the change in the system.

Thus this chapter presented the result of a systematic literature review and taxonomy of impact analysis that was conducted on existing approach used in software systems and addressed in detail the existing methods used for assessing modifications in enterprise systems and business process management in general. This literature highlights three main limitations of the current approach as follows:

- There is no evidence indicating the method for change management to monitor the modification of ERP system during the post implementation phase. All the approach and technique that discussed in the literature is for a change management during the implementation phase of the ERP systems.
- A number of techniques for analysing the impact have been discussed in the literature while these techniques provide an excellent example of how to apply in the specific domains, it can be difficult to implement them to assess the modification of ERP system with complex integration.

- The current change impact analysis approaches are not able to address the implications of change for on-going transactions in the system. Which means that during the modification, the system may face with hundred instances operating in the system and during the implementation a business analyst require a technique to monitor these instances in such a way to terminate the operations safely with less impact.

So our approach tends to focus on the impact of proposed changes at the operational level, in order to close these gaps. The method draws a parallel between ERR post-implementation change management and engineering change management in product design and manufacturing. As in the case of engineering change management in product design, ERP changes should be carefully assessed before being implemented, to evaluate their impact on the current structure of the system, and possibly predict effort and plan the implementation phase.

CHAPTER 3 RESEARCH METHODOLOGY

3.1 Overview of Design Science Research

A pragmatic approach to scientific research involves the use of a method that appears to be best suited to the research problem. Quantitative, qualitative and natural science research method is commonly adopted in scientific research, the one approach that specifically addresses gaps that exist in many academic types of research, particularly in the managements and information system disciplines are the design science research method. The design science research is an enhanced complementary methodology of the more common behavioural science research paradigm as it produces clear contributions to knowledge-based in the form of contracts, model, method and instantiations.

According to (Wieringa 2014), the two major components in a design science research project are the design activity and the research activity. The design activity builds on the understanding of the context to develop innovations that support a specific need whereas the research activity seeks to understand the interaction between the artefact and the context, e.g., to investigate the needs of stakeholders, or to evaluate a deployed artefact. Design science is an iterative process, and Hevner refers to the interplay between the design activity and the research activity as the evaluation loop (Hevner et al. 2004) The two activities alternate during repeated iterations in the loop until the results are satisfactory.

The research adopts the Design Science Research (DSR) paradigm (Hevner et al. 2004). DSR has emerged in the last 15 years as a legitimate approach to management information systems and industrial engineering research. It is based on the implementation of a wide-range of sociotechnical artefacts, e.g., modelling tools or decision support systems, to address issues clearly recognised by practitioners using and possibly extending well-established scientific theory and principles. The DSR for an Information System (IS) incorporates principles, practices, and procedures required to carry out Information System research and meets objectives (Peffers et al. 2007). Design research is inherently a problem-

solving process that creates and evaluates IT artefacts intended to meet business needs and solves the identified organisational problems (Hevner et al. 2004).

In DSR (see Figure 3-1), the research is influenced by the "Environment" (people, organisation, technology), which defines the relevance of the research work for practice and academia, and by the "Knowledgebase" (foundation and methodologies), which provides rigorous theories and principles for the research. The "Research" involves the design and development of theories and building of artefacts, justification and evaluations. The research eventually gives something back to both Environment (e.g. a solution) and the knowledge-based (e.g. new insights).

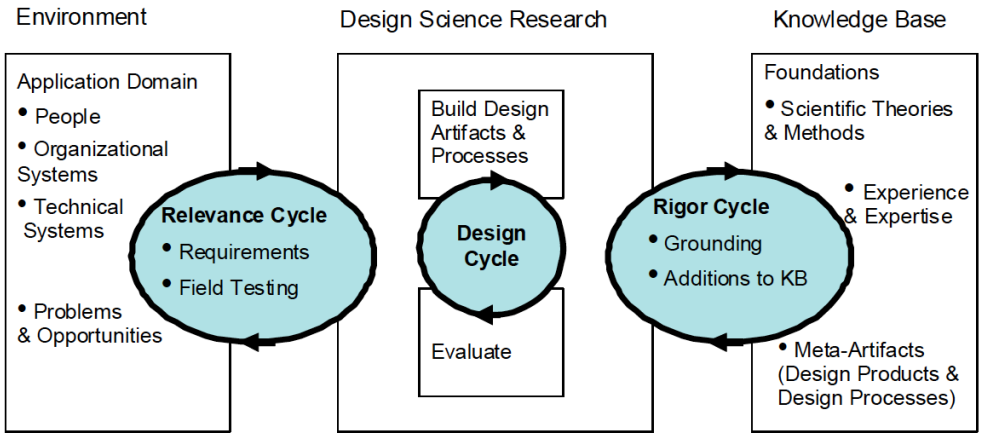


Figure 3-1 DSR framework (Hevner et al. 2004)

3.2 Adaption of Design Science Research

Figure 3-2 provides a graphic interpretation of the DSR framework developed by (Hevner et al. 2004) customised to the context of this project. The research need is driven by the difficulties encountered by organisations in managing ERP post-implementation change, as highlighted in Chapter 1 Section 1.2. To solve this issue and to achieve the objectives delineated in Chapter 1 Section 1.3, this research work develops and evaluates the following artefacts:

1. A methodology to identify the different phases of controlled ERP post-implementation change management;

2. A dependency meta-model of ERP system components that can be applicable for mapping dependency relations;

3. A taxonomy of possible ERP post-implementation changes;

4. A set of mechanisms (algorithms) to evaluate the ripple effects of the proposed ERP changes, based on the dependencies identified by the dependency meta-model;

5. A set of business intelligence metrics to assess the impact of the proposed modification;

6. A software tool (decision support system) embedding all the conceptual artefacts defined above, to assist business analysts in change management.

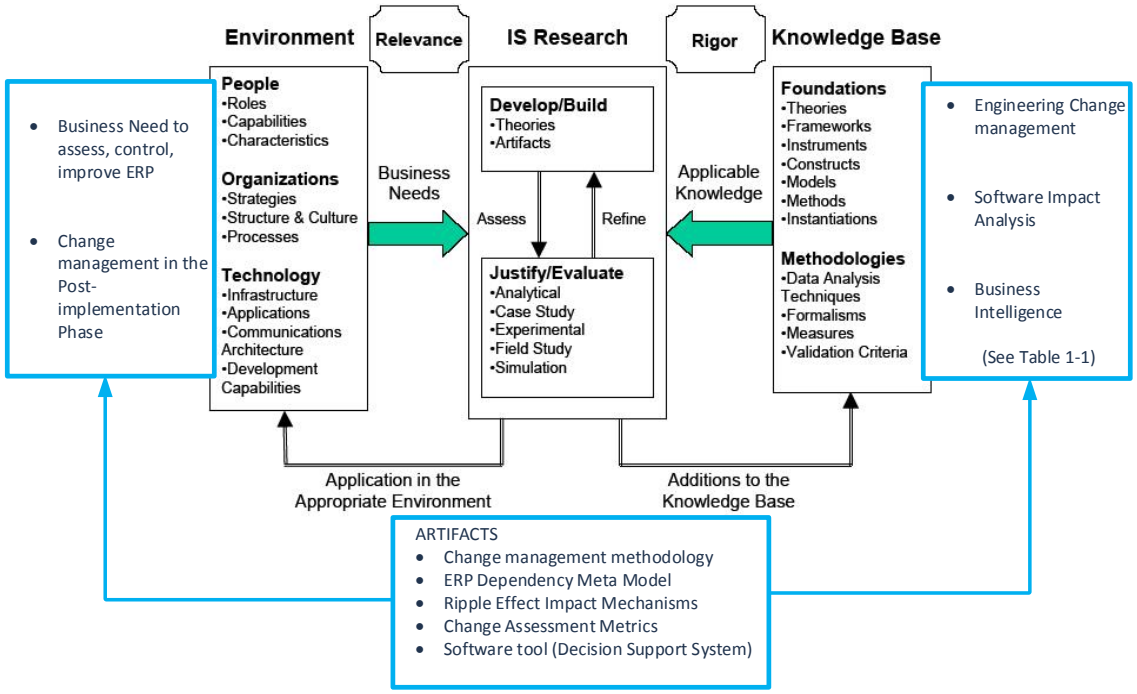


Figure 3-2 DSR framework contextualized to this research (Hevner et al. 2004)

The design and implementation of the artefacts listed below will draw from a vast knowledge base, as shown in Table 1-1. Knowledge-based provides the raw materials from and is composed of foundations and methodologies. Methodologies provide guidelines used in justifying and evaluating phases. Rigour is achieved by appropriately applying existing methodologies and foundation listed below.

Table 3-1 Knowledge base of the research

Applicable knowledge	Application area (Brereton et al. 2007)
Engineering change management	Change management methodology specific to ERP post-implementation
Software impact analysis, workflow evolution, service-oriented architecture evolution.	Principles to analyse ripple effects of modification in large software system (workflow-based and service-oriented)
Business analytics	Design of a set of assessment metrics
Model-driven software engineering	Design and implementation of a software tool (decision support system)

3.3 Research Road Map and Phases

The process model consists of six constructs representing activities that should be carried out during a Design Science research. The activities include problem identification and motivation; define the objectives for a solution; design and the development; demonstration; evaluation; and communication.

The activities we follow in this research are as follows:

1) Problem identification and motivation: We identify the target domain for our research the ERP system change management during Post implementation phase, the specific research problem, the main requirements, challenges and the value of the proposed solution. Section 1.3 in Chapter 1 stated the importance and relevance of the problem that this thesis pursues to solve.

2) Define the objectives for a solution through the study of Literature Review: Systematic literature reviews are important for different reasons (Brereton et al. 2007): (i) to summarize existing evidence concerning a practice or technology, (ii) to identify where there are gaps in current research, (iii) to help position new research activities; and (iv) to examine how far a given approach is supported or contradicted by the available empirical evidence. We explain how the new artefacts (i.e. framework) is expected to help and provide a solution for the problem. We analyse some of the existing research in change management for ERP post-implementation and the methods for impact analysis of modifications in enterprise

systems. Then references the findings to specify our design strategy for developing the ERP change management framework.

3) Conceptualization and formalisation: We develop the new framework, which involves determining the artefacts desired functionality and architecture to develop actual artefacts. During this phase, we develop a generic conceptual meta-model of ERP system to determine the dependency among ERP system components. Then we define mechanisms in order to assess the impact of the different type of change, and lastly, we identify a set of metrics to estimate the depth of impact during ERP system modification. The resulting artefacts from our research are described in Chapters 4-5.

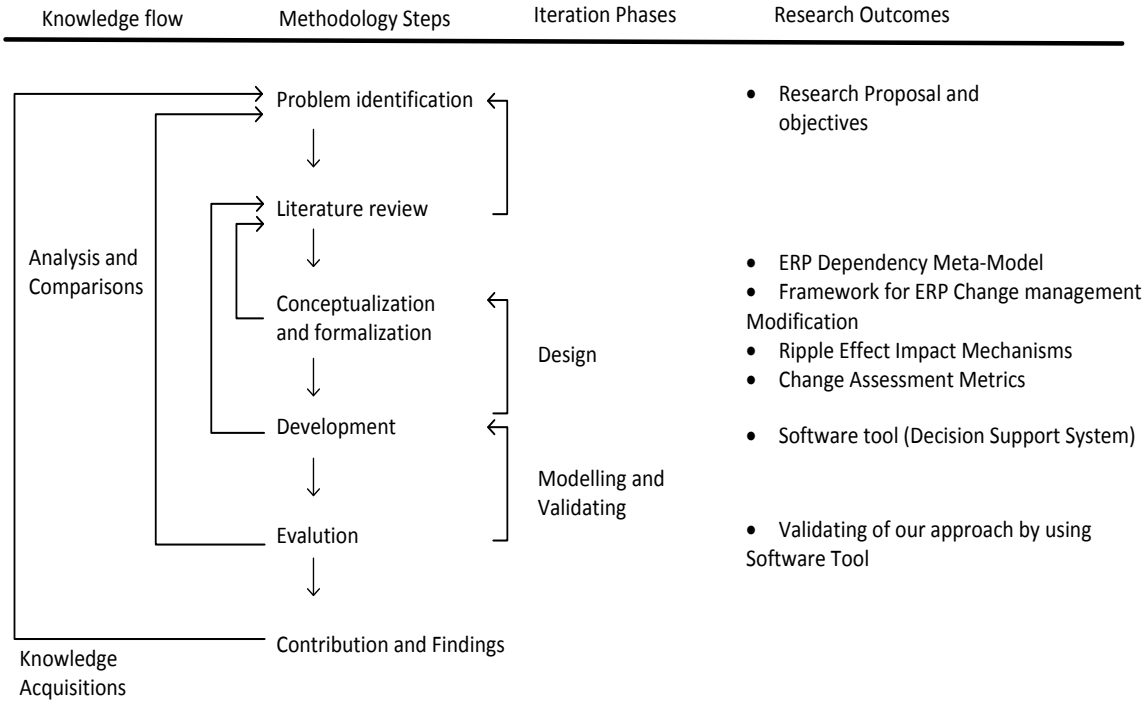


Figure 3-3 Research Iteration Phases

4) Development of Software tool and demonstration of case study: we implement a software tool, i.e. a decision support system embodying the identified models, method and metrics. We use prototyping to prove our concept and apply the proposed framework in the

context of ERP change management where the process of change implemented for two case studies of ERP system.

5) Evaluation: we observe how the artefacts support a solution to the problem we examine the proposed framework and tool by looking at its capabilities and evaluating the fitness of our ERP change management framework. We compare the result with our objectives by using the context in the demonstration. We use standard design science research evaluation method presented in (Hevner et al. 2004) where we follow the descriptive approach by conducting specific scenarios around the developed framework to demonstrate its utility. Furthermore, we use qualitative analysis for the evaluation. Chapter 8 provides empirical evidence that validates the achievement of the objectives of this thesis by evaluating the resulting artefacts.

6) Communication: Based on the result and contribution of this research the application of our approach should improve change impact analysis and reduce the risk associated with ERP post-implementation change management. We communicate the findings and contribution of this research with our peers in the forms of conferences and journal publication. A list of publications is provided in Chapter 9 "conclusions."

3.4 Research Validation and Analysis Method

The author (Shaw 2002) identified types of research validation in software engineering: analysis, experience, example, evaluation and persuasion. The most common kind of validation is evaluation through empirical research methods: observational methods (e.g. case studies or field studies) or experimental methods (e.g. controlled experiments or simulation).

Evaluation methods are based on a set of criteria to validate the claims about artefacts to be evaluated (Hevner et al. 2004). He lists artefact criteria, which can be applied to assess the quality of design science artefacts. These criteria, which also coincide by March/Smith, are the functionality completeness, consistency, accuracy, performance, reliability, usability, and fit with the organisation. The criteria the researcher use depend on the reason that they developed the artefacts in the first place. They will be related to the original research

questions. The evaluation can lead to conclusions about the design process as well as the design product and may suggest that further modifications to either or both are needed.

As far as the evaluation of the artefacts developed in this thesis is concerned, to increase the external validity of the results, this thesis will adopt a combination of different assessment methods. The proof-of-concept implementation of the software tool can be considered as the evaluation of the feasibility/implementability of the proposed methods and tools.

Other aspects, such as the "usefulness", i.e., the fit-for-purpose of the proposed framework to solve the problem for which it has been devised, and the "ease of use" of the framework are evaluated with case studies (usefulness), expert opinion (usefulness and ease of use), and controlled experiments (ease of use). The activity should evaluate "how well the artefact supports a solution to the problem".

Our methodology has been devalued in the individual session with ERP expert from different organisation and discipline. For evaluation, we create case studies of ERP post-implementation change using the simulated teaching organisations of two large packaged ERP systems, namely SAP and Microsoft Dynamics NAV. SAP is the worldwide market leader in ERP technology, while Microsoft Dynamics products are challenging SAP's lead particularly for medium enterprises in the US and European markets. We have access to both systems in this project through their respective academic alliance programs.

We evaluated the framework, through the software tool and case studies, with a panel of ERP experts, i.e., business analysts and solution designers from large consulting companies. We provide two different types of evaluation of our tool. First, to show its feasibility in practice, we evaluate to what extent our tool can handle standard functionality provided by commercial ERP package. This evaluation activity has involved a panel of 7 ERP professionals with average 10 years above experience of the ERP system in the different industry. Then we presented the result of an experiment with ERP users to assess the fit for purposes and usability of our tool in practice. This evaluation has involved the panel of 7 ERP professional and 12 master students in the business system who recently completed an introductory course on business process management and enterprise systems.

The expert has been chosen because of their experience in a range of different ERP software package in various industries. Finally, to evaluate the usability of the methodology and the software tool, we ran controlled experiments in simplified scenarios using students with a background in business process management and enterprise systems as participants.

The research methodology chapter presents research processes that are employed towards answering the research questions posed in Chapter one. These include the analysis procedures used for conducting the literature review, constructing and evaluating the artefacts (the tool for the impact analysis framework of EPR modification).

3.5 Summary

In this thesis applying the design science research methodology as a framework provides the researcher to validate information systematically, and at the same time to ensure that the research is scientific and delivers valid results. This chapter explained the design science approach and presented the procedure of the proposed methodology and adaptation to our research problem. The procedure in this method consists of six constructs representing activities that should be carried out during a Design Science Research. In the sub-section of this chapter, the research roadmap is introduced which explains the phases of the approach and iterations. The chapter ended by describing a brief summary of research evaluation and analysis method to increase the external validity of the approach and the designed artefact.

CHAPTER 4 FRAMEWORK DESIGN PART 1 (PROCESS AND CONCEPTUAL DEPENDENCIES)

4.1 Introduction

ERP systems are likely to be changed because of both external and internal purposes. A particular change of an ERP system could make various levels of impacts, which mainly depend on the types of dependencies among the different elements of the ERP system. As discussed in Chapter 3, a significant amount of research on change management has been done in the context of organisational change and ERP system adaptation, suggesting that the modification of the ERP system should be regulated and monitored through a procedure in order to verify the impact of any change in the system. Numerous techniques have been conducted to support the impact of modification on software systems, but there is no known technique supporting the evolution in the specific case of ERP systems. In this chapter, we present our approach to support the assessment and evaluation of change impact in the specific case of ERP systems. Existing approaches to change management of software systems served as a guide to building a framework for this research. The change process considered in our framework, in fact, incorporates the impact analysis taken from two traditional approaches, i.e., Structured Analysis and Design Technique (SADT) of (Bohner 1996) and the Engineering Change Request of product lifecycle management from (Wright 1997).

The description of the framework is divided into two chapters to facilitate readability. This chapter (Chapter 4) reports the overall change process of our framework and presents a comprehensive analysis of ERP dependency relationships and ERP change requirement analysis. The next chapter (Chapter 5) focuses on the impact analysis mechanisms and impact assessment of the estimated effort for ERP modification.

This chapter is organised as follows; we first present the proposed change process framework of generic ERP system modification in Section 4.1 followed by an overview of each phase in a sub-section during this procedure. From Section 4.2 of this chapter, we

provide a complete definition of each artefact that has been designed to be used in our framework. Section 4.2 explains the ERP dependency meta-model artefact for defining the dependency relationship between ERP components. Finally, Section 4.3 concludes this chapter by discussing the taxonomy of ERP modification and the use of change template for analysis of the requirement. To conclude this chapter, we provide an example of a new requirement to show the deployment of each artefact related to the phase 1 and 2.

4.2 ERP Change Process

The design and utilisation of the ERP product, as previously discussed, is subject to change in the post-implementation phase, due to changing business requirements, availability of new technology, or bug fixes. Therefore, modification of existing features in the ERP system or change into the organisation's requirements is the most common maintenance activities after, post-implementation of ERP systems. An ERP, in fact, can be seen as a complex product in use by the entire organisation. When a modification is made in particular components of the ERP system, it is hard to determine the impact on the rest of enterprise components due to the complexity of the design. For example, changing a data object, such as "purchase requisition", may require changing the function that creates the purchase requisition object and can propagate to all functions that use the data object as an input parameter. Besides, at the same time, many business cases of a different type of purchasing may be on-going in an ERP system as a reflection of organisational operation. To what extent these business cases are affected by the proposed change and whether their execution can terminate safely after implementation of modification is also a concern that has to be managed. Business analysts and developers have to tackle the problem of change propagation and to trace the entities in order to change the ERP system.

A number of methodologies and techniques for analysis of the impact of the change in source code, workflow system, business processes, and service-oriented system have been discussed in the literature (Lehnert 2011a). While all provide an excellent example of how to apply impact analysis in a specific domain, it can be difficult to implement them to assess the modification of ERP system, which are complex systems encompassing an intertwined set of process, functions, data and so many running instances.

Our methodology draws a parallel between change management and traditional engineering change management. To start analysing the impact of modification in the system, we first need to understand the process of change (Wright 1997) (Jarratt 2004). Figure 4-1 shows a model of generic engineering change process based on (Jarratt 2004). On the same process, we show that our methodology for change management in ERP focuses on the late "before approval" stages, i.e., impact assessment of a proposed change solution, and provides decision-making support in the "during approval" phase.

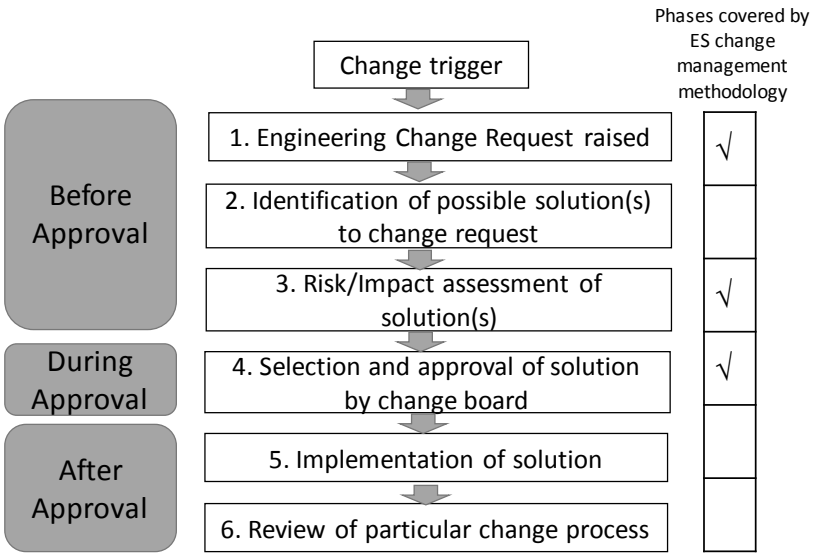


Figure 4-1 A model of a generic change process from (Jarratt 2004)

Our framework also simplifies the steps identified by (Bohner 1996) in software system and employed other methodology in change management as mentioned in above. So to assess the impact of modification during ERP system post-implementation the framework in Figure 4-2 is presented. This framework comprises four phases. Each phase exploits one or more artefacts, which is used to evaluate the ERP modification. Figure 4-2 also demonstrates the sequence and interaction of each phase following the related artefact for each activity. These four phases jointly form the change process of an ERP system that contains as:

- Phase 1: Understanding ERP system dependencies
- Phase 2: Requirements elicitation and analysis;
- Phase 3: Impact analysis and assessment;

- Phase 4: Modification implementation.

The main feature of our framework is to analyse and assesses the effect of modification through system dependencies so that the process of change can be run faster and more efficient. After the ERP has gone live, we first need to understand and map the dependencies among ERP components as pre-requisition steps in order to manage change and their impacts. So, when the need for change occurs (i.e. the organisation recognises that a modification is necessary), the change process starts by applying phase 2, 3 and 4 to complete the change process. Essentially, the dependency relationship should be updated each time the ERP system has agreed to change.

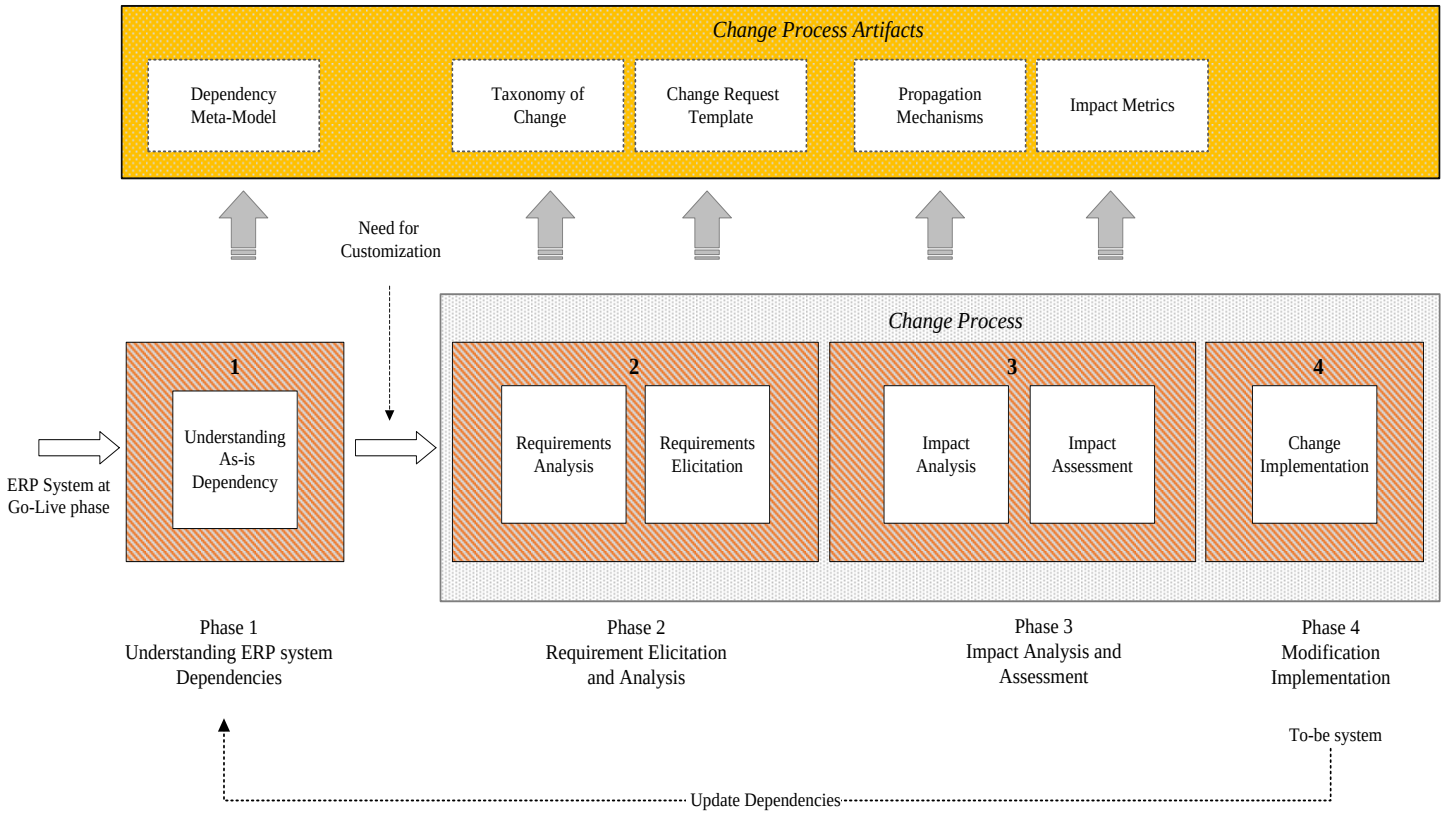


Figure 4-2 Framework for ERP modification

The following sub-sections explain in more detail these activities during the process for each phase separately. Note that this thesis focuses only on the analysis and assessment of the ERP modification, that is, it does not focus on the actual implementation of the change

in the system. In other words, the last phase of Figure 4-2 (Change implementation) is out of the scope of this thesis.

4.2.1 Phase One: Understanding ERP System Dependencies

Software development naturally produces a blueprint of the product structure such as design drawings schemata or material specification, which are used to assess the impact of the requested changes, in terms for instance of design feasibility and cost (Hamraz, Caldwell, and Clarkson 2012). Normally, the implementation of ERP systems does not necessarily produce a similar blueprint if the dependencies among ERP system components, which an analyst can use to assess the impact of changes during post-implementation. Hence, mapping the conceptual components of the ERP system and their dependencies is the first step towards assessing the impact of future post-implementation changes.

A modification can start from any of the components (Business process, Business function or data object) and propagate to other elements according to the dependencies. During the modification procedure capturing and analysing the affected components are time-consuming and costly if handled manually. This is due to the various dependency relationships across the ERP system components at different levels of ERP enterprise model. Therefore, as noted in the previous section, the initial phase of the modification process is to specify the dependencies between ERP system components.

There are two advantages of specifying a model of ERP entities dependencies. First, it enables the change identification and analysis procedure to be automated and faster, therefore, it is more efficient than the manual. Second, we can define a more accurate result because of the well-defined dependency representation while manual identification may lead to some affected entities remaining unidentified because of errors, or missing relationships.

Dependency relationship between ERP system components is captured based on the ERP dependency meta-model that contains all set of components at the enterprise level such as business processes, business functions, business data, etc. The dependency meta-model of ERP system is designed in the form of a UML class diagram in which usually each ERP component denotes as an entity and each association lines as a dependency relationship between entities. Dependency relationships are usually represented in the form of graphs or

tables that assist the business analyst in understanding the system specifications (Dai et al. 2009). Every time that the change agreed for implementation this dependencies relationship must be updated according to the new requirements and their dependencies so that the new change request can capture the impact propagation correctly and accurately based on updated dependencies.

A design of our meta-model follows from considering a traditional 3-tier architecture of ERP system (Manuel and AlGhamdi 2003, Smets-Solanes and De Carvalho 2003) the GUI, application and data layer. The ERP architecture model allows to understand the ERP system components at different layer and to define how the ERP system operates. Later in this chapter, Section 4.3 provides a detailed description of the ERP dependency meta-model.

4.2.2 Phase Two: Requirement Elicitation and Analysis

As in the case of traditional engineering change management (Wright 1997) change requirements have to be documented in a way suitable to undergo the implementation review phase. Systematic development and redesign of products, systems, or software start with requirements elicitation and analysis.

Requirement elicitation concerns the discovery of change requirements, which involve the user when the change is triggered by a change in customer requirements. The elicitation is supported by change taxonomy that identifies the possible type of changes. When business analysts want to specify the proposed change request to the ERP system, they need to determine which parts of the system they are asking to change. For this reason, the taxonomy of possible changes of an ERP system is developed to help in classifying the "scope" of the change request. This change request can be described by a set of change criteria (Lee 1998). According to the taxonomy of change (Lehnert and Riebisch 2012), the new requirement transfers to the set of change primitives that the impact analysis mechanism can understand. As a rule of thumb, change type can be derived from the Create, Update, and Delete primitive operation.

Requirements analysis is the activity by which a new requirement is described as a change request. A change request is a document containing a proposal for an improvement of an ERP system; it is of primary concern in the change management and change process. A

request is declarative, i.e. it states what needs to be accomplished, but leaves out how the change should be carried out (Laudon and Laudon 2000). A particular change request may translate into a combination of changes to the ERP components applied in a certain order or can only one component. This critical phase should clarify the modification task, which allows to understand the objectives and goals and to determine the particular needs and conditions of developing a proposed modification in the ERP system.

During the change process, a change request must be made through the standard template, described in Section 4.4. In engineering change (Eckert, Clarkson, and Zanker 2004), this step is usually captured by a change request template that is a standard template to capture change requirements in a suitable way for the review team. The change template defines the core requirements of the ERP modification request. The business analyst creates the change requests that involve understanding the needs of the end-users; the contexts where to-be-developed; modelling, and documenting the need for change.

4.2.3 Phase Three: Modification Analysis

This phase is concerned with the analysis of the requested change, in order to decide whether it is worth implementing. According to (Bohner 1996), Impact analysis evaluated the ripple effect of the proposed change. In engineering change management, the execution of this phase is highly unstructured and involved a change committee in charge of reviewing the effects of the proposed change (Hamraz, Caldwell, and Clarkson 2012). In our methodology, given the dependency meta-model, this phase can be largely automated. This is part of the main methodological differences between the ERP post-implementation change management and traditional change management. The modification analysis involves two sub-phases, i.e. impact analysis and impact assessment.

4.2.3.1 Impact Analysis

Impact analysis determines the scope of the modification and what other components are needed to be modified for applying the change. This shows the complexity of the change component to other associated components in the system, i.e., if the change affects a sufficiently large part of the static structure of the ERP system or a large portion of ongoing operations. One of the most important areas in an enterprise systems environment is the

capability to predict where and how new requirements impact the existing system (Lindvall 1998). Identification and analysis of dependencies became more challenging in such environments compared to traditional monolithic software systems. This is because enterprise systems such as ERP system are perceived as complex software products embedded in the complex organisational environment and to define the impact of modification requires a systemic and structured analysis.

One of the core processes in redesigning a system is to capture undesirable side effects, which affect other components apart from the targeted one (e.g. change in function may impact on the business processes using this function). The effects can cross different layers of the enterprise model. Besides, the modifications do not just affect the design structure but also concerns the active instances (i.e. business cases) during the system execution. For example, a modification of a data object in the ERP system can potentially impact the design elements like functions that are associated with the data object and also it may impact the instances of data objects that are created due to the execution of the related function.

Here in this phase, impact analysis concerns the automatic detection of dependencies by developing a set of algorithms through the utilisation of dependency model and taxonomy of change from phase one and two. This phase is accompanied by definition of a set of change impact mechanisms, i.e., algorithms that can be run automatically to calculate the impact of the different type of change identified by the change taxonomy. The mechanisms operate when an ERP component is modified or added to the system that results in a list of items to be redesigned.

The methodology draws a distinction between impact analysis list at design-time and run-time:

- Impact analysis at design-time concerns identifying the ripple effects (i.e. impact) of change on the static structure of an ERP system determined by the dependency model. This part of the analysis is important for all ERP systems.
- Impact analysis for run-time concerns identifying the effects of the change in the ongoing operations of an organisation that the ERP is supporting at the moment the change becomes operational. This part of the impact analysis is applicable for

ERP systems supporting long-running operations of an organisation. In this case, changes may affect running instances of long-running business processes. Note the run-time impact analysis should also propose a solution to safely terminate the running instances affected by the change in case the is implemented.

The impact of a new requirement is analysed and predicted through the execution of the impact algorithms by impact committee during a change process. The predicted impact is expressed regarding the changed of enterprise system components i.e. business process, functions, and data object. The explanation of how this mechanism operates is detailed in Chapter 5 of this thesis.

4.2.3.2 Impact Assessment

Impact assessment is concerned with estimating the magnitude of the change and, possibly, the effort required for its implementation and its potential costs for the organisation. The quantitative effects of that change on other ERP impact items are the major concerns of the study of impact analysis. To understand the software concerning the modification, we must determine parts of the ERP system that may affect the change and review them for possible further impacts (Bohner and Arnold 1991). Our methodology supports the impact assessment sub-phases by providing a set of impact assessment metrics to quantify the design-time and run-time impact of a proposed change. These can be used to apply traditional decision-making techniques to support business analysts in their decision about accepting the change. Once the impact was predicted and the effort estimated, the changing community extensively used the results for impact propagation, cost estimation, and project planning (Lindvall 1998).

In engineering change management, the planning phase provides how the modification goal will be achieved. During the planning stages, the business analyst and ERP developer suggest a set of feasible implementation strategies taken from the study of ERP customization. At design-time, this concerns estimating the cost and effort of different options for implementing the proposed change, e.g., in-house code modification versus bolt-on to a service of an external provider (Rothenberger and Srite 2009). At run-time, planning

concerns identifying policies for safely completing the execution of all run-time running instances affected by the change.

A choice of overall modification strategies is important because it can have a significant impact on the outcome of ERP systems changes. Poor decisions at this stage can have enormous cost implications for implementation of the modification. For example, there is a significant difference between bolt-on strategies and ERP code modification strategies for implementing new functionality in the system. Chapter 6 presents a research study that highlights these differences between implementation strategies based on the opinion of ERP experts and Chapter 5 presents how to use this result to plan the modification effectively.

Once the change has been approved, the change needs to be implemented. As discussed before, the implementation of change does not simply mean to modify the enterprise systems as requested but also concerns the management of active transactions in the ERP to terminate in a safe state. That is a state in which execution of the enterprise can smoothly continue after the change has been implemented. Instruction on how to implement change is defined in this phase so that the next step knows where and what is require to implement the change.

4.2.4 Phase Four: Modification Implementation

The change implementation phase takes the final change operations and executes them in the ERP system. This phase concerns the actual implementation of modification on the impacted items in the ERP system were identified in phase three, which can be applied according to the selected strategies during the planning activities. This needs to be done based on the user's preference after the impacts of the change operations are reviewed and approved (Abgaz 2013).

The implementation modification can be achieved by a different method like configuration as a simple strategy that entails on choosing appropriate components of ERP system and settings the parameters according to the ERP reference model, or more advanced approach such as customization of existing component through code modification of ERP system. An ad-hoc extensions or modifications of the ERP package also are concerned as an external solution as another alternative to support the ERP systems modification, such as

bolt-on features, or ERP programming. Various ERP vendors are purchasing third party solutions by tailoring their ERP system with an extra module or functionality. In some cases, the third-party solution may require additional programming code to interact with the ERP package like interface development solution. Alternatively, if any of the above solutions are not appropriate, the exact requirements of the organisation implementing ERP can be met through direct modification of the ERP package source code. This option usually implies a significant amount of effort and specialised expertise and remains code modification an issue when the system is upgraded to a new version, as the modified code parts may not update correctly.

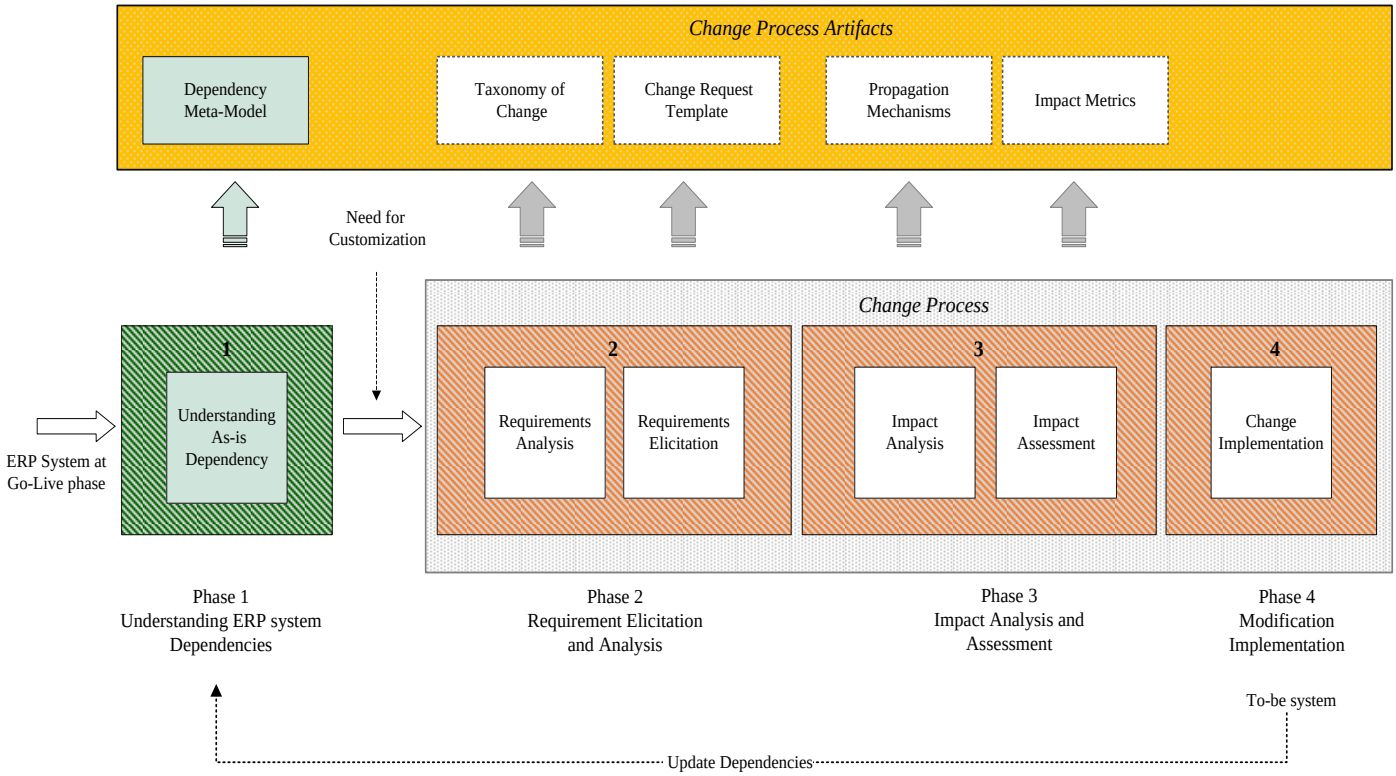


Figure 4-3 Change Process as Phase 1

As mention in Section 4.2, a set of artefacts needs to be created in supporting change process of our framework. The artefacts here are the products of each phase that emerges when a change management team engages in running the analysis of modification. Following

sections describe these artefacts in more detail and explain the relevant use of each artefact during the change process (See Figure 4-783).

4.3 ERP Dependency Meta-Model

Identifying dependencies in ERP systems is essential to ensure adequate change management process. Poor understanding of such dependencies may lead to higher maintenance costs and catastrophic effects on business operations. On the other hand, dependency analysis in the ERP system is challenging, as dependency relationships exist in a distributed and heterogeneous environment. Therefore, we require a meta-model to identify the dependency relationship. Numerous ERP systems have been introduced in the market based on the concept of modular structures, n-tier architecture, and centralised databases. A typical ERP system has n-tier model consists of at least three layers of database, application layer and, the interface layer (Smets-Solanes and De Carvalho 2003, Manuel and AlGhamdi 2003).

In Database layer, data connections are used to attain data from a different data source based on adapters. In application layer, the external and internal resources are located and invocated together (Cai, Bu, and Jiang 2012). The controller in application layer refers to BPM engine that could execute BPEL, BPMN, EPC or other business process execution language. In application layer, a fully adopted ERP system involves in various business processes each of which poses a sequence of procedures within a functional department (Wang and Xu 2009). An end-to-end business process begins with its starting function, proceeds one by one function or sub-process, and ends with its finishing function (Johannesson and Perjons 2001). The next layer is the user interface that can be either a custom program or a web browser by which the end user interacts with the system. On the user interface layer, the user could use services to personalize their work environments so as to meet the business requirements.

Further to what discussed before about the 3-tier abstraction of ERP architecture, we consider the typical architecture of ERP systems into four levels as defined in Figure 4-4. The user interface level where the end users interact with the system, the business process level that contains a set of end-to-end business process. The business function level contains

a set of functionalities and feature that operate during business process execution. This tier serves as an example of services from service-oriented architecture in the application logic layer that interact with business processes (Winter and Fischer 2006). The example of business functions is order processing, order planning. Furthermore, the business functions use business data to complete the task. Also, lastly, the business data level where the input and output of each functionality maintain and store in the database layer.

In the representation of Figure 4-4, each level uses only the level directly below it, for instance, business functions using the business data as input and output but not the other way around, or business function cannot use the business process but business processes using the business functions to operate.

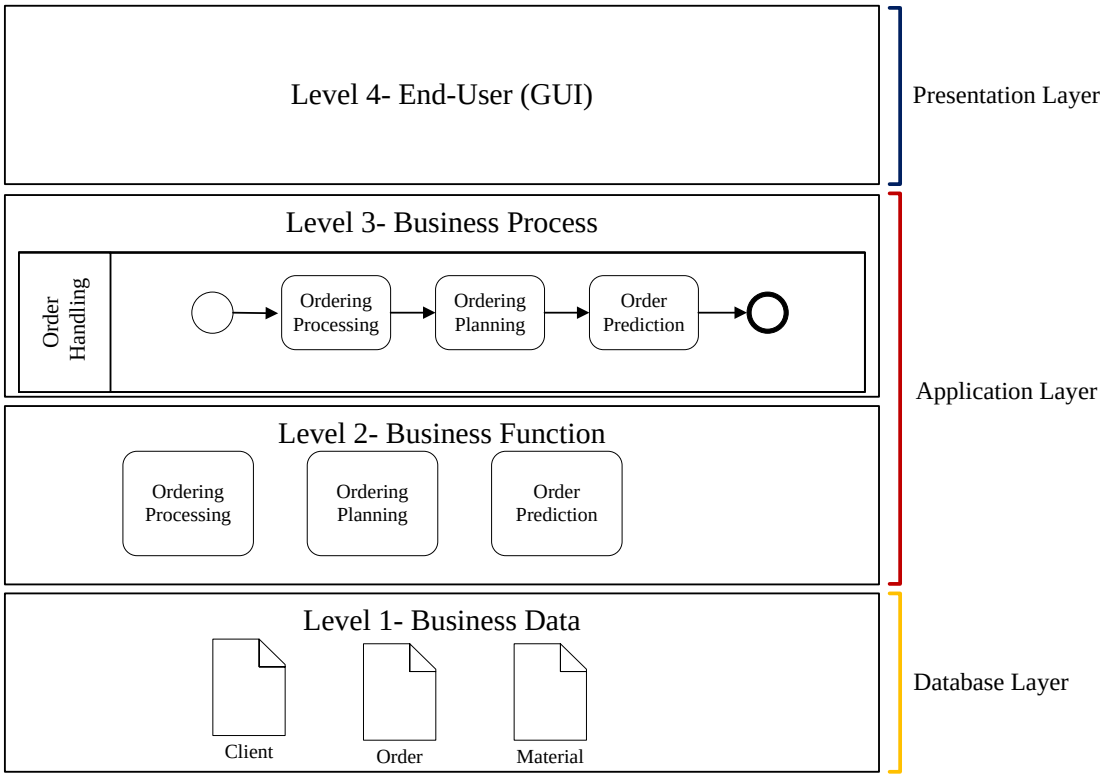


Figure 4-4 Four Tier of ERP Enterprise Model

The requirement for the utilisation of dependencies for change impact analysis is the need for expressing dependencies. Based on the architecture of Figure 4-4, we can now define the ERP dependency meta-model as one of the main artefacts that underpin our

impact analysis framework. In Figure 4-5, the ERP dependency meta-model shows the representation of the entities and interactions of the ERP components to define the dependencies relationship. The description of ERP system entities and relationship is explained as follows:

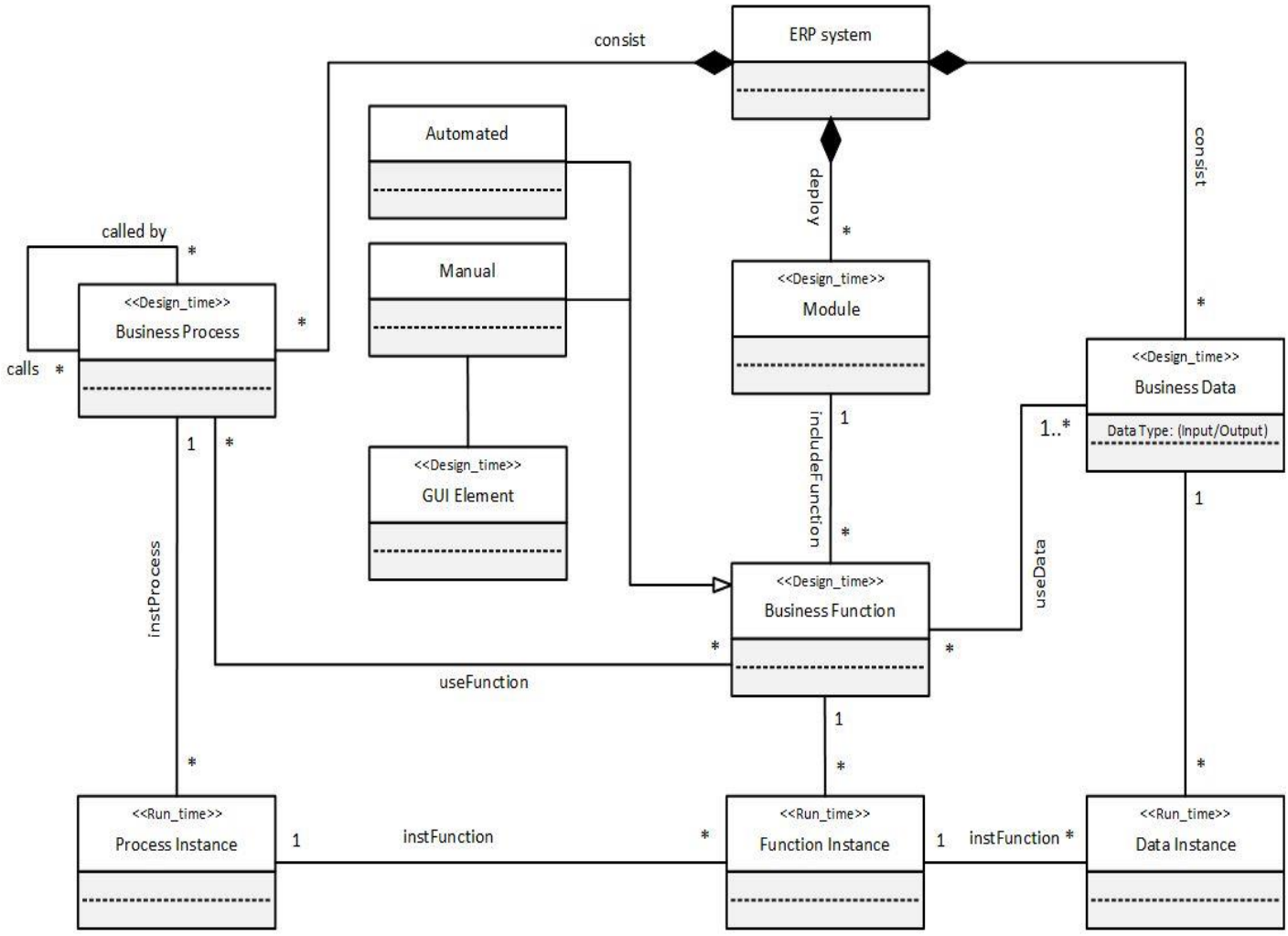


Figure 4-5 ERP Dependency Meta-Model

Business Data: a business object is a collection of attributes stored in the centralised database. Business data may refer to master data, such as customers, materials, vendors or transactional data, like purchase order, sales orders, etc. During the ERP systems execution, a business data is manipulated by standard CRUD (create, read, update and delete) functionality to maintain and store data in the system.

Business Function: A function is an operation that manipulates one or more business data. A function retrieves information from one or multiple business data as an input, e.g. to create a sales order information is captured from the customer requisition and possibly creates/updates one or more business data as an output, e.g. sales order are modified by different functions along the sales process. To be more precise, functions constitute the smallest unit of work, which usually requires human or machine resources for their execution. Functions may be automated, e.g. updating inventory during material shipment, or can be performed manually, which requires the ERP end user to enter data value. This performs through GUI element (i.e. desktop browser or mobile device). Functions can be internal, i.e. implemented within the ERP package, or they can be available in an external system in the form of add-on or "bolt-on" functionality with a compatible interface.

Module: ERP systems are composed of various modules. Each module specialises in a particular business area of an organisation. Modules are individually purchased based on organisational needs and are typically responsible for aggregating and processing information for a separate business function or departments or a group thereof.

Business Process: ERP systems implement a collection of business processes. A process orchestrates different functions from one or more modules in the system. A Process model can be composed; i.e., business function, as well as sub-process, exposing another business process model (Aldin and de Cesare 2011). Processes often run through the cross departments, which provide a coordination mechanism in the business. For example, the sales order process is to be incorporated into the payment process and the production process to fulfil an order from the customer.

ERP systems are information systems that support the integration of business processes of enterprises. In ERP system, the integration of business process occurs when the business process calls another business processes during the execution. The execution of the process model is dependent on other processes is known as sub-process, which represents a step in a process model. Every time a sub-process enables during process execution, it corresponds to the execution of another process model. A sub-process has input and output data containers to pass data between it and the subordinate process. Therefore, the business processes dependency in the ERP system is described as two types:

- Called by: The execution of the business process is triggered by the execution of another business process.
- Calls: The completion of the execution business processes depends on the completion of other business processes.

Runtime: The dependency meta-model of Figure 3-4 also comprises entities needed to capture the status of an ERP system at run-time, i.e., while business processes are executing. Once an executable process model has been deployed on a process engine, new process instances can be created and executed according to this model (Reichert and Weber 2012). Several instances of the same process model may exist representing different business cases. The process engine employs a state model to control the concurrent execution of these process instances; i.e., each process instance presents an internal state representing its progress toward completion and its status with its functions and data objects. When the preconditions for executing a particular function are met during run-time, a new instance of this function is created (Reichert and Weber 2012). Hence, a function instance represents a single invocation of an action during the execution of a particular process instance. Furthermore, function instance utilises data related to its corresponding process instance, and itself produces data used by succeeding functions. When executing data object (an electronic form) attributes of the corresponding object instance may be read, written, or updated.

4.3.1 Run-time and Design-time

As depicted in Figure 4-6, a distinction is made between the design time and the runtime environment. Design environment allows the business analyst and developer to verify and configure different perspective of an executable process model in the ERP system. As far as the run-time is concerned, process instances represent the individual execution of the business process, e.g., the processing of a specific purchase order. Several business transitions known as the instance of the same process type may exist exposing different cases in the system (Choudhury, de Cesare, and Di Florido 2008) (e.g., purchase order process of different material). Each process instance exhibits an internal state representing the progress toward completion. A newly created process instance is described as active or enables, which then all the functions of the process model become enable as well.

Process instances create instances of individual functions, which may require the instantiation of specific data objects, or “documents”, for their execution. So when the preconditions for executing a particular function are met during run-time, a new instance of the function is created (Reichert and Weber 2012). A function instance represents a single invocation of an action during the execution of a particular process instance. Further, function instance utilises data related to its corresponding process instance, and itself produces data used by succeeding functions. When executing, business data (an electronic form) attributes of the corresponding data instance may be read, written, or updated.

Once the new process instance starts the execution, the status will change from active to running state, which indicates the progress of the process instance according to the state of function instances in the process model. When the status of all function instances turns into completed, the process instance transfers from running to completed.

Figure 4-6 shows the example of a process instance representing purchasing process of different orders. The example demonstrates three different cases of process instance state of inactive, active and completed. Instance I1 indicate as inactive where all functions and data objects instances become inactive at the initial stage to create a new purchase order.

Then the instance I2 shows the active state of a process, which demonstrates a combination of all functions instance state. The state of functions indicates the progress of business process instances during the execution such that instance 2 shows the function related to the maintaining quotation has been completed and the next function creates a purchase order is enabled. The last instance of the example shows that all the pre-condition and post-condition of all function instances are accomplished so that the process instance can transfer to the completed state.

Moreover, Figure 4-6 demonstrates the relationships between the function instance and process instance. When a process instance is inactive or completed, the state for all function instance associated with process instance also becomes inactive or completed respectfully. However, when the process instance is active, there is always a case in which one of the function instances is in the active state. The rest of the function instances can be in the state of inactive or completed.

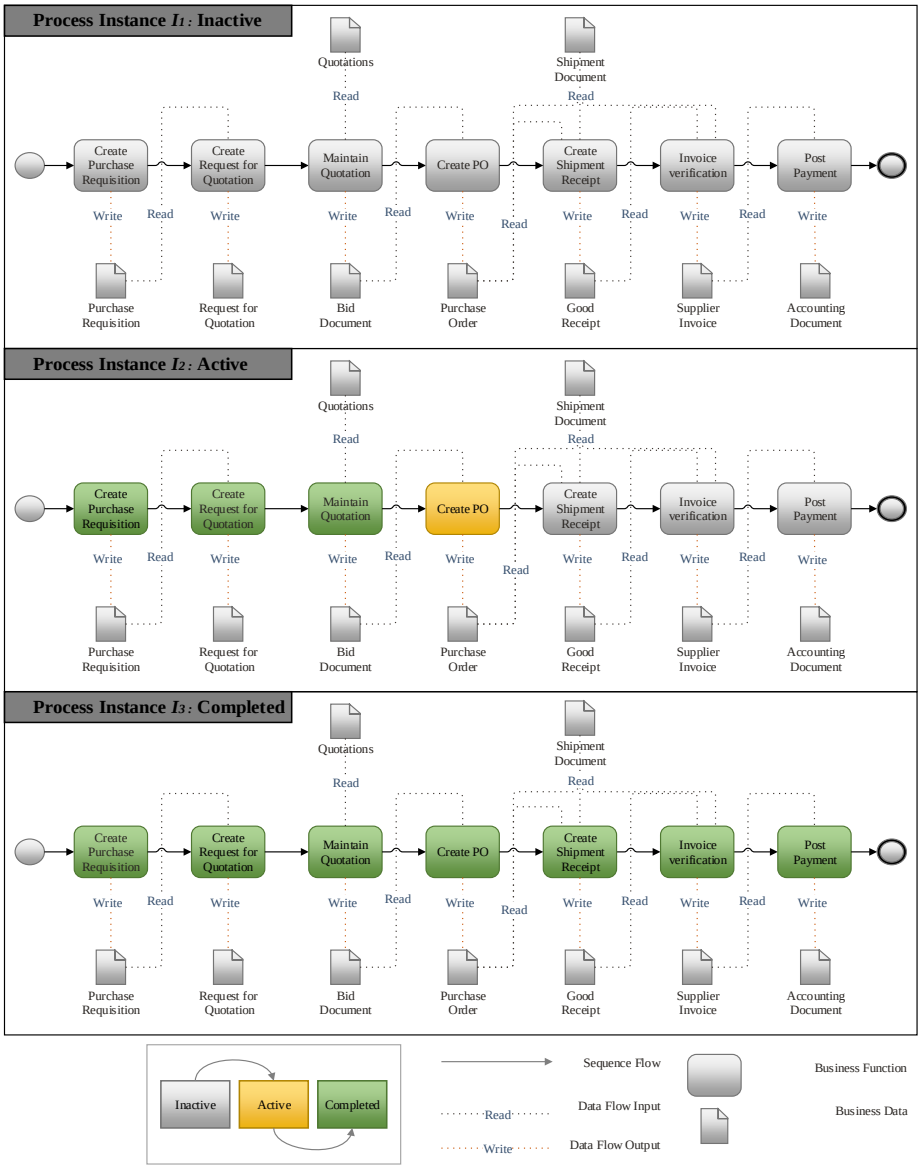


Figure 4-6 Example of Running instances

4.4 Taxonomy of Change in ERP Systems

As (Wright 1997) indicated that in the engineering change management change request have to be documented in a suitable way to undergo for reviewing the impact analysis and implementation. Understanding the new requirement is an essential part of the change process to translate them into a set of change primitive for determining the impacts. The change taxonomy result from the application of the primitive create, delete, and update to the design time components of the dependency meta model (Lehnert and Riebisch 2012).

This section investigates how change operations can be modelled and classified to allow for automated change impact analysis (Lehnert and Riebisch 2012). A change operation, in short, " a change", transforms the system or one of its constituents from version n to version n + 1. In order to describe and model changes, the following information is required: the component that is going to be changed, and then the description of the change activity on a particular component. Taxonomy of ERP modification in this section represents as an artefact, which supports the Phase 2 in our framework depicted in Figure 4-7.

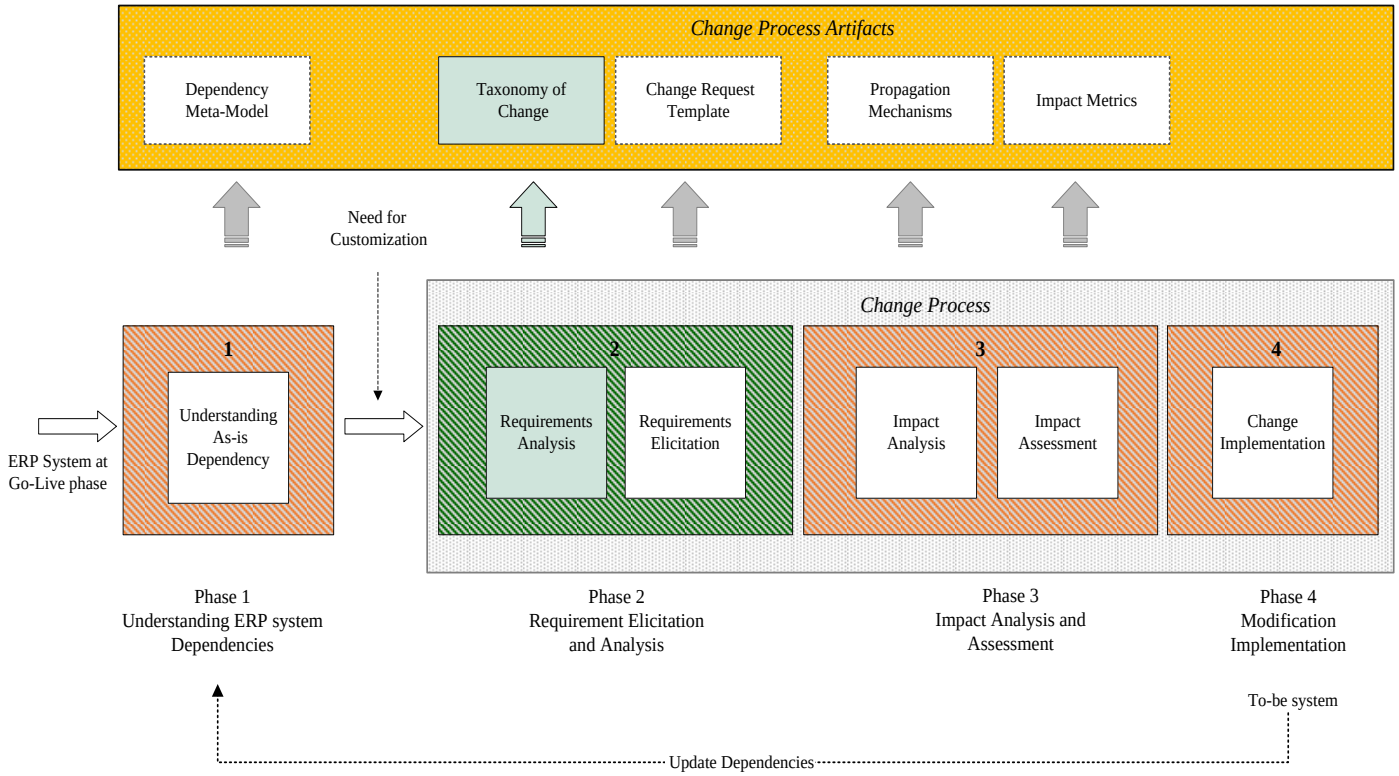


Figure 4-7 Change Process at Phase 2

The following discussions are based on a literature survey on the modelling and classification of changes in the fields of impact analysis. This classification also is derived from the study of ERP misfit literature that defines the gap between an organisation requirement and ERP system capability into the business process, function and input and out misfit. There are different types of change that have been studied in the enterprise systems

and business process management literature (Reichert and Weber 2012). We classify ERP modifications along three dimensions, i.e. the level, type of change and the granularity of change operation.

4.4.1 Granularity: Atomic and Composition Change

To describe granularity of change activities, (Fluri and Gall 2006) introduced the concept of basic (or atomic) changes. As the name suggests, atomic changes describe change operations that cannot be further refined or broken down into other changes. (Fluri and Gall 2006) List adds, delete, and update as a set of basic change operations. This concept was extended by (Lehnert and Riebisch 2012) who introduced the notion of composite operations that are comprised of sequences of other atomic operations. The definition of granularity of change in the context of ERP modification change process is represented as follows:

- Atomic Change: a change activity that is comprised of exactly one non-interruptible operation. Each atomic operation can only involve a maximum of an entity in the ERP system from dependency meta-model.
- Composite Change: a change activity that is comprised of at least two atomic or composite change operations. A composite operation may involve at least two or more of the entity in the ERP system from dependency meta-model.

The concept of atomic and composite operations is most suitable for serving as a solid base for the modelling of change operations for impact analysis tasks. In our approach, composite operations may consist of sequences of either atomic or other composite change operations.

4.4.2 Level and Type of ERP Modification

The level of change is determined from the dependency meta-model entity on which the change occurs. The approach, therefore, distinguishes ERP modification at the level of business objects, functions, and business processes. Figure 4-8 is embodying the taxonomy of change artefact for defining and classifying the change request. In this illustration, change taxonomy of ERP post-implementation results from the application of the primitives creates, delete, update the design elements business data, function and process identified by the dependency meta model in the previous section. Note that a module in an ERP system is

simply a homogenous collection of functions and business processes related to one specific part of an organisation's value chain and, therefore, modules are not subject directly to post-implementation changes. Note also that in our instantiation we do not consider changes to GUI elements. GUI elements are usually modified to address issues of usability (Lucas and Babaian 2012). Since our work focuses on post-implementation changes to address changing business requirements, we consider these changes out of scope.

The type of change specifies how the change occurs. It is important to identify the type of change (Kherbouche et al. 2013) that our change process can support between adding (creating), deleting or upgrading an entity in the ERP system, e.g. updating a function or deleting a business object.

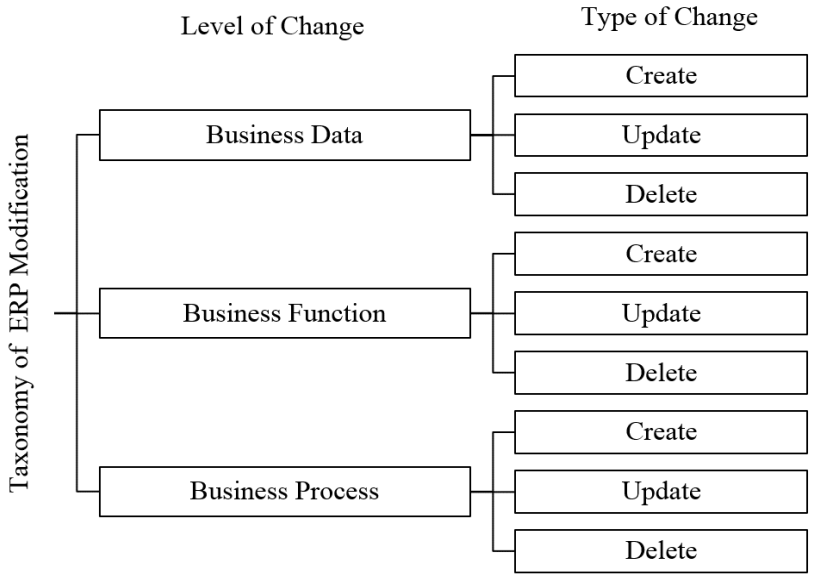


Figure 4-8 Taxonomy of ERP Modification

To exemplify the above classification, suppose a scenario in which an organisation requests to upgrade the function when creating a purchase requisition document. The upgraded function should enable users to be informed about the latest price supplied by the vendor when making a purchase requisition. This type of modification is considered to be the update (type) of a function (level) that known as the atomic change request. In our taxonomy of change, we consider the updating of the business process as a change operation.

4.4.2.1 Updating Business Processes

Change operations in a business process model can be described as addition, delete, update, and move/replace of the business function or modification of the execution sequence business process. The example of ERP Business Process modification refers to the research of impact analysis of business process section from a literature review by (Weber, Rinderle, and Reichert 2007b) in order to monitor business process modification. In this research (Reichert and Dadam 1998, Weber, Sadiq, and Reichert 2009) they address the taxonomy of change and the impact of modification on the business process in the information system. Table 4-1 presents an overview of these 14 types modification scenario taking to account when updating a business process from p version to the new version p'. Each pattern is under a name and a brief description that include the change in business function or the change in the execution of the process.

Table 4-1 Taxonomy of Business Process Modification (Reichert and Weber 2012)

Change Pattern	***Name***	***Description***
P-1	Insert Business Function	An existing Function is added to the process
P-2	Delete Business Function	A function is deleted from process
P-3	Move Business Function	Function move from current position to another in process
P-4	Replace Business Function	A function is replaced by another function
P-5	Swap Business Function	Two existing function are swapped in the process
P-6	Copy Business Function	Using same function in the process twice
P-7	Extend Sub Process	Business Function are extracted and replaced by corresponding sub process
P-8	Inline Sub Process	A Sub process is dissolved and directly embedded in the parent process
P-9	Embed Business Function in loop	Add a loop construct to a process
P-10	Parallelize Business Functions	The functions which have been confined to be executed in sequence are parallelized in a process
P-11	Embed Condition	An existing function shall be only executed if certain conditions are met
P-12	Add Control Dependency	An additional control edge for synchronizing the execution order of two parallel
P-13	Remove Control Dependency	Remove Control edge in the process
P-14	Update Condition	A transition condition in the process is updated

4.5 Change Request Template

This section explains the artefact related to the requirement analysis activity as described in Figure 4-9. After the classification of change in ERP system is defined the requirements of end user need to be translated according to the taxonomy of change and dependency model.

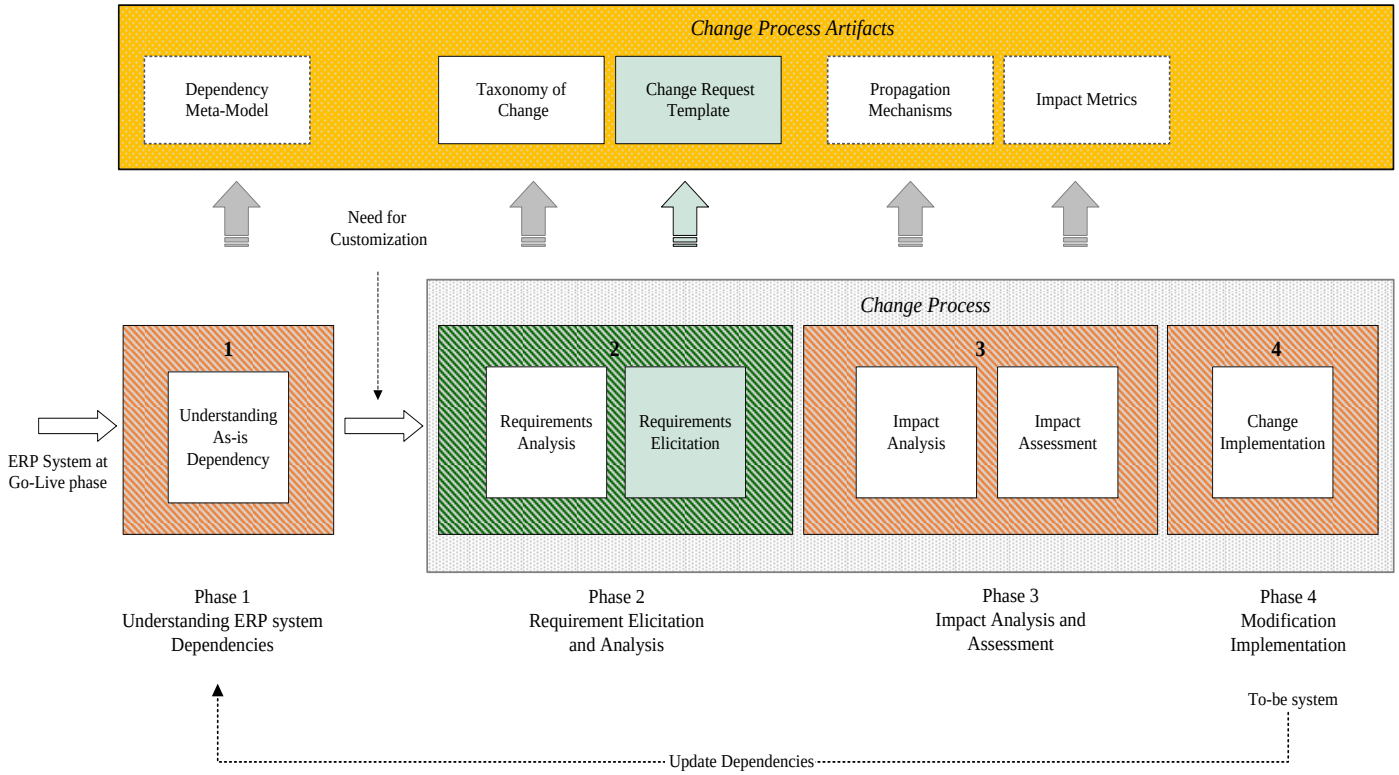


Figure 4-9 Change Process at Phase 2 (Requirement Elicitation)

The template in Table 4-2 captures the basic requirement to proceed with the changing scenario before applying the modification in ERP systems. Each change scenario has a unique code following by date, title and a brief description of the end user requirement.

The change request template is passed on to the business analysts as submitter who is completing the form by analysing the requirements. The business analyst has a comprehensive understanding of ERP system capability and compares the request with ERP system specifications. Then, translate the changing scenario into a set of change requests.

Change request operates on single components of the ERP model (e.g., to add or remove a business function, or to modify the workflow of a business process) that can be run in the impact analysis tool to identify the effects. Next, change requests are prioritised according to the dependencies identified by our dependency meta-model of ERP systems. For instance, if the change concerns implementing a new functionality, then it is essential to make sure the input(s) and output(s) of the function, so that the function can execute without any problem. Otherwise, if current business data are not suited for the new function, additional change requests then added to the list as a pre-condition for the function. So, the impact analysis first checks the data as the pre-condition and then run the impact mechanism for the function.

Table 4-2 Change Request Template

Change Request Guideline	
Element	***Description***
CR-Code	The exclusive Code When Change Request was created
Date Reported	The date the Change Request was created
Requested by	Assigned by the ERP end-user (Consultant)
Title	A brief description of the change request
Submitter	Name of the person completing the CR Form and who can answer questions regarding the suggested change
Description	Description of the desired change and how the change Should works
Atomic Change	A list of basic requirements ('i.e. Atomic Change) in order to implement the change request
Priority	A code that provides a recommended categorization of the urgency of the Primitive Change from Extremely important to Less important
Status	- Approved - Pending for Approval - Pending for Impact Analysis - Rejected
Impact Summary	A list of number of components that impacted as result of change request

The change request template is passed on to the business analysts as submitter who is completing the form by analysing the requirements. The business analyst has a comprehensive understanding of ERP system capability and compares the request with ERP system specifications. Then, translate the changing scenario into a set of change requests.

Change request operates on single components of the ERP model (e.g., to add or remove a business function, or to modify the workflow of a business process) that can be run in the impact analysis tool to identify the effects. Next, change requests are prioritised

according to the dependencies identified by our dependency meta-model of ERP systems. For instance, if the change concerns implementing a new functionality, then it is essential to make sure the input(s) and output(s) of the function, so that the function can execute without any problem. Otherwise, if current business data are not suited to the new function, additional change requests then added to the list as a pre-condition for the function. So, the impact analysis first checks the data as the pre-condition and then run the impact mechanism for the function.

After the business analyst defines and prioritises the change requests, the impact analysis tool is used to identify the effect of the proposed modifications. The output of the impact analysis is then used to complete the "Impact Summary" section of the template. Each change request form has a statue that represents the stage of change request procedure. Approved, rejected, and pending for approval is the stage where the request is submitted for approval according to the impact analysis results. At this point, the request form presents the summary of impact components as a consequence of the modification that is aggregated from analysis of atomic change.

4.6 Example of Change Scenario

In order to show the application of the phases and artefacts of our framework presented so far, this section discusses an example of a possible change scenario involving the modification of an ERP system. As far as stakeholders are concerned, four different groups of stakeholders are involved, i.e. the business analyst, impact committee, ERP end-user, and ERP developers. Figure 4-10 matches the activities of the change process as discussed in our framework to the stakeholders.

ERP end-user usually trigger the need for change due to an error detected during the execution or a need for new functionality. This new requirement can be issued from one person (e.g., a purchasing consultant) or an organisational group (e.g., a purchasing group). Then the new requirements of the ERP user, (i.e., system administrator, operation staff, or ERP end-users) are captured in the form of a request for impact analysis.

The Business Analyst, as an individual or a group of individuals that are familiar with the ERP system specification and organisation requirements, evaluates the change request

and translate the new requirements into a set of change primitives identified by our framework (i.e., the taxonomy of change of Section 4.4). They are in charge to propose a plan for implementation of the modification according to the system configuration and appropriate implementation strategies.

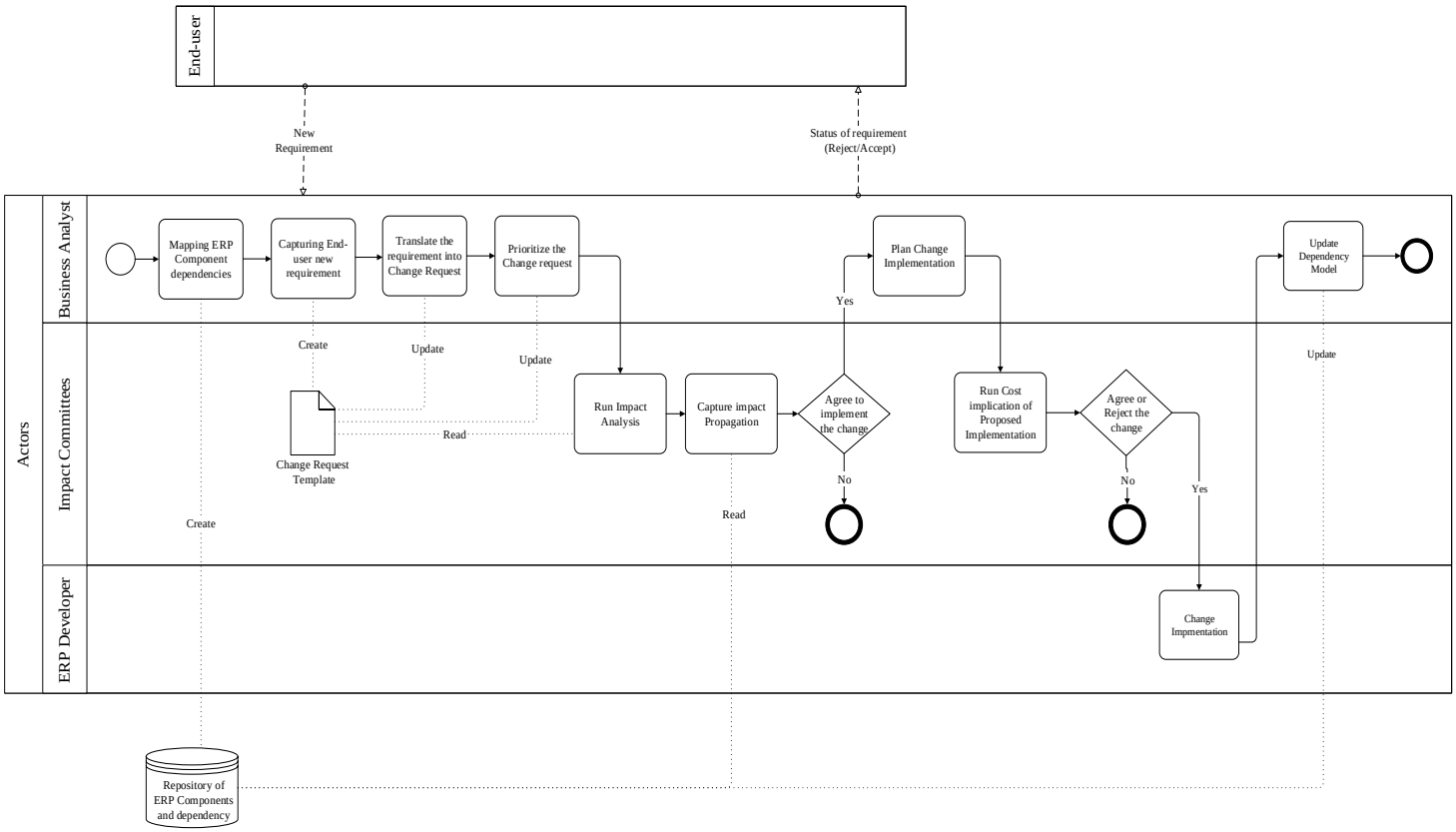


Figure 4-10 Change Process of ERP Modification

An impact committee is a group of people who analyse the impact of ERP modification and are familiar with on how to operate the impact analysis. In addition, this committee also in charge of decision making based on the result of impact analysis on whether to accept or reject the change request.

Upon the acceptance of the change request the committee asks the business analyst to evaluate the impact and provide the planning strategies for implementation. Then the committee estimates the cost implication of implementation strategies and makes a decision on the implementation. They also allow rejecting the change request if the cost impact of implementation is considered as high risk and expensive to perform. Once the request agreed

for implementation, the ERP developer applies the change according to the proposed implementation plans and adjusts the ERP system.

In the remainder of this section, to exemplify the overall process and the activities explained in this chapter an example of change scenarios is provided. The example involves a manufacturing company requesting to improve the purchasing process by enhancing the purchase requisition functionality. Purchasing is the typical process of buying materials or services from suppliers (vendors), as represented in Figure 4-11. Usually, the process starts with creating a requisition that defines the requirements for the material or service.

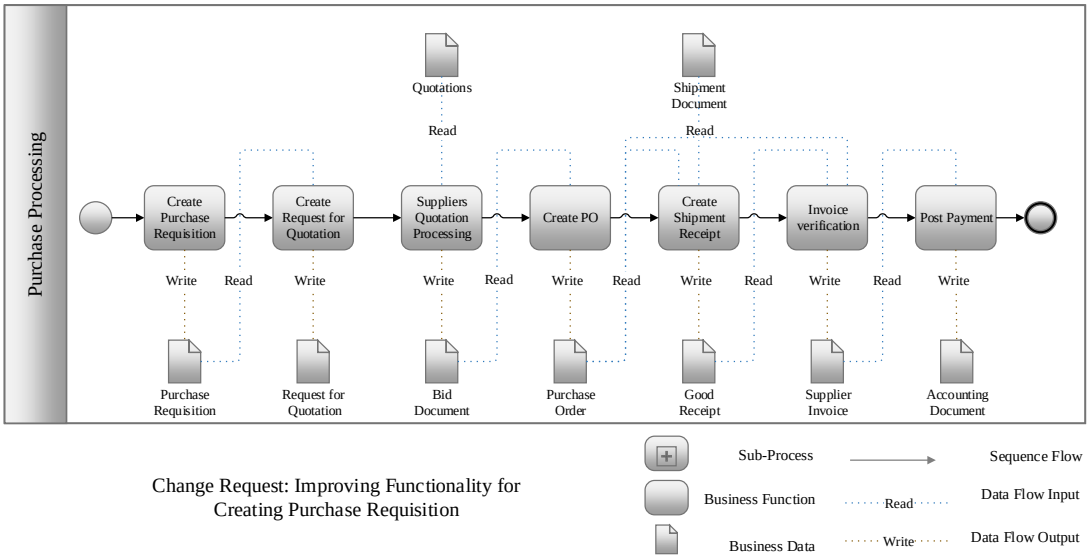


Figure 4-11 Example of Purchasing Process

The request for quotation based on the requisition document is then sent to the suppliers to be able to provide the material or service. Suppliers post their quotations in response to the request. Then, the purchasing group has to review the quotations and select the best offer based on the price, availability, delivery and quality of the offer. Normally companies purchase the same material from various suppliers, so they have a history of purchasing materials, commonly known as purchase records.

In our scenario, the company requests to develop the functionality while creating the requisition, to determine and calculate an alternative price based on the purchase history. This functionality requires an enhancement of the purchase requisition document (business

data) with an extra field showing an approximate range of the price that is automatically obtained from purchasing history. This information allows the purchasing group to understand the approximate price of purchasing material and also to inform the suppliers to submit the relevant proposal within the price range price suggested in the requisition document.

Table 4-3 Example of Change Request

Change Request Guideline	
Element	***Description***
CR-Code	CR-001
Date Reported	10-October-2015
Requested by	Purchasing Consultant (E. Willams)
Title	Improving Functionality of Creating Purchase Requisition
Submitter	J. Anderson (ERP Business Analyst)
Description	The Motorbike Manufacturing Company requested to improve the functionality of creating purchase requisition. The functionality should be extended to provide an estimate of alternative prices of the product/service to be purchased by searching the purchase history of the company. This information enables the purchasing group to understand the approximate price of the purchase and also defines the range of price for their suppliers.
Primitive Change	Update Business Function (Creating Purchase requisition) Update Business Data (Purchase Requisition)
Priority	CR-001-1: Update Business Data (Purchase Requisition) CR-001-2: Update Business Function (Creating Purchase requisition)
Status	- Pending for Impact analysis

In the change process, it is fundamental to define the type of change to begin the evaluation of modification requests. Table 4-3 shows an example of a change scenario expressed using this template in the case of updating a specific function (Creating Purchase Requisition) in the ERP system.

To do so, the business analyst first translates the above scenario in a set of change requests. The next step is to prioritise the change requests based on the order of implementation that is driven by the dependency meta-model. In our case, to perform modification at the function level, it is important to ensure that the required business data exist in the system so that the function can run without any problems. Otherwise, if there is a need to modify business data, it is first required to analyse the feasibility of change at that

level and then proceed to the functional level (see the priority assigned in Table 4-3). After prioritisation and classification of change requests, the impact analysis can start running.

4.7 Summary

As outlined before this the framework specification of our work is split into two chapters for readability purposed. This chapter focused on the overall change process of our framework and presented a comprehensive analysis of ERP dependency relationship and ERP change requirement analysis. First, a standard change process for ERP systems modifications was given. Then, provided an overview of the artefact involved in the change process. Finally, this chapter demonstrated in detail a first subset of the artefacts defined in our framework, namely the ERP dependency meta-model artefact for mapping the dependency relationship of ERP components, the taxonomy of ERP modifications and the ERP change request template. Furthermore, the chapter concluded by demonstrating an example of a new requirement to show the deployment of each artefact related to the phase 1 and 2. The next chapter focuses on the impact analysis mechanisms and impact assessment of the estimated effort for ERP modification.

CHAPTER 5 FRAMEWORK DESIGN PART 2 (IMPACT ANALYSIS AND ASSESSMENT)

5.1 Introduction

Impact analysis determines the scope of the modification and the complexity of the change. The impact assessment has the capability to predict where and how modification affects the existing components in the system (Lindvall 1998). As mentioned the impact analysis concerns automatic detection by capturing the impacted items through the utilisation of dependencies. The mechanisms operate when an ERP component is modified, deleted or added to the system that results in a classification of items to be affected by the change (i.e. data objects, functions, processes, instances, etc.). Once the impact is predicted and the effort estimated, the result extensively is used for defining the impact propagation, cost estimation, and implementation planning.

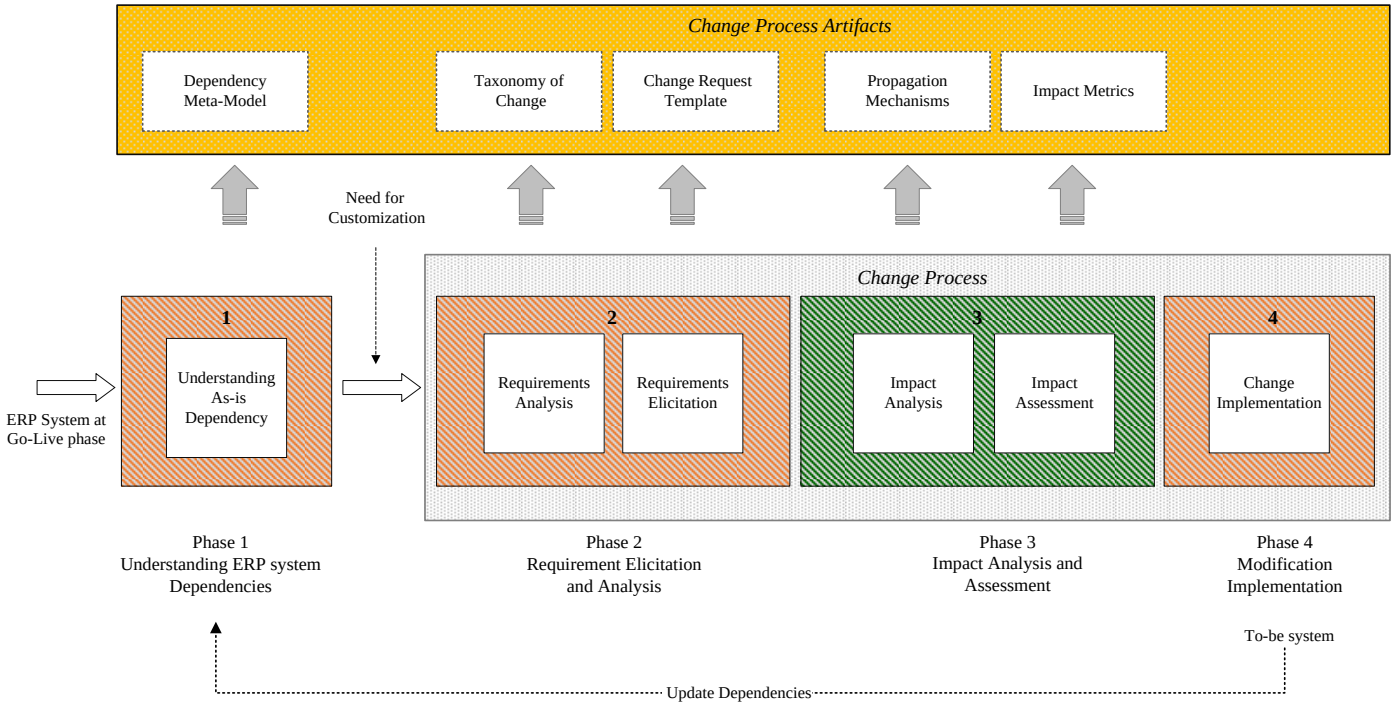


Figure 5-1 Change Process at Phase 3

This chapter deals with the second part of our framework, refers to impact analysis and assessment (Phase three in Figure 5-1). The chapter is divided into two sections that explain the artefacts (i.e. impact analysis and impact assessment). The initial part describes the impact analysis and how the mechanism is performed to capture the affected item in our context. In the second part of the impact assessment will be discussed. The impact assessment enhances decision-making to plan the implementation of the modification efficiently.

This chapter is organised into six sections as follows: Section 5.2 Provides an overview of the impact assessment by specifying the requirement for developing impact mechanisms. As for Section 5.3 defines what compatibility entails in our context compared to the various use and overloading of the term "compatibility" in software system literature. Section 5.4 describes the techniques for managing modification during the execution time of the ERP system. In Section 5.5, we present mechanism artefacts to analyse the impact set of proposed changes. Section 5.6 presents the impact analysis metrics that we developed as a guideline for decision making.

5.2 Preliminaries to Design Impact Analysis Mechanisms

ERP system evolution requires a comprehensive approach to be able to manage the new requirements that result from upgrading, deleting or creating an ERP component during the post-implementation lifecycle. Two fundamental concepts need our attention before formulating the impact analysis mechanisms:

1) The notion of compatibility: The definition of compatibility is used for evaluating whether a new version of an ERP component, as specified in the change template, can substitute an existing one or not. Compatibility is important to assess the design-time feasibility of the proposed change.

The ERP system consists of several interconnected entities, as mentioned in Chapter 4. When the change process requires update one of the components of an ERP system to a newer version or replacing it with an ad-hoc solution (e.g. by a third party application). So it is important that the proposed changes be compatible with the other existing components of

the ERP system, to prevent the risk of interrupting the operation of the ERP system during and after the change process. Incompatible modification might affect groups of components in ERP systems differently, according to the dependency relationship with other entities. Therefore, the primary goal of this chapter is to introduce the activities of impact analysis for assessing the impact of modification on the ERP system.

2) The notion of Critical Point (for running instances affected by the change): The critical point, provides a way to handle and regulate the proposed change in the instances of the process (es) affected by the change that is still running in the system.

As far as the runtime is concerned when a modification is requested, different actions can be made to handle the changes at runtime based on the execution state of the process instances. In particular, a notion of a critical point is required in a process instance during impact analysis. Our impact analysis suggests one strategy for each process instance that is more appropriate to implement a change according to the execution state. The following next sections 5.3 and 5.4 explain each concept in a comprehensive form.

5.3 Compatibility Among ERP Systems Components

This section discusses the notion of compatibility among ERP system components in the context of our research. First, it explains the reasons for focusing on the notion of compatibility during the modification process. Then it describes existing techniques to assess the compatibility among components of the ERP system. Note that the analysis of compatibility assessment and measuring the compatibility value is out of the scope of this thesis and this section aims only at giving an overview of the techniques that must be employed during the activity of ERP modification to assess compatibility.

Given the diverse use and overloading of the term “compatibility” in software system literature, we define what compatibility entails in our context. Typically, compatibility addresses the evolution of ERP system by considering the zero impact on the applications components when implementing changes in the ERP system. Meaning that, a new ERP system is compatible with an existing one if the use of the new component has no (zero) effects on the rest of the ERP system as compared to the utilisation of the existing component.

The notion of compatibility is used to compare two ERP components based on the description of the component's interface and behaviour. This notion can determine whether one component can be substituted for another or not. For instance, referring to the example in Chapter 4 by applying the modification to the function in the ERP system, it is essential to first measure the similarity of the new function against the old one and to ensure that the execution can perform without problem.

For assessing the compatibility, we introduce different notion of compatibility for each level of change based on the study of various literature (Bellahsene, Bonifati, and Rahm 2011) (Yan, Dijkman, and Grefen 2010) (Dijkman, Dumas, and García-Bañuelos 2009) (see Figure 5-2):

- Compatibility Assessment for Business Data
- Compatibility Assessment for Business Function
- Compatibility Assessment for Business Process

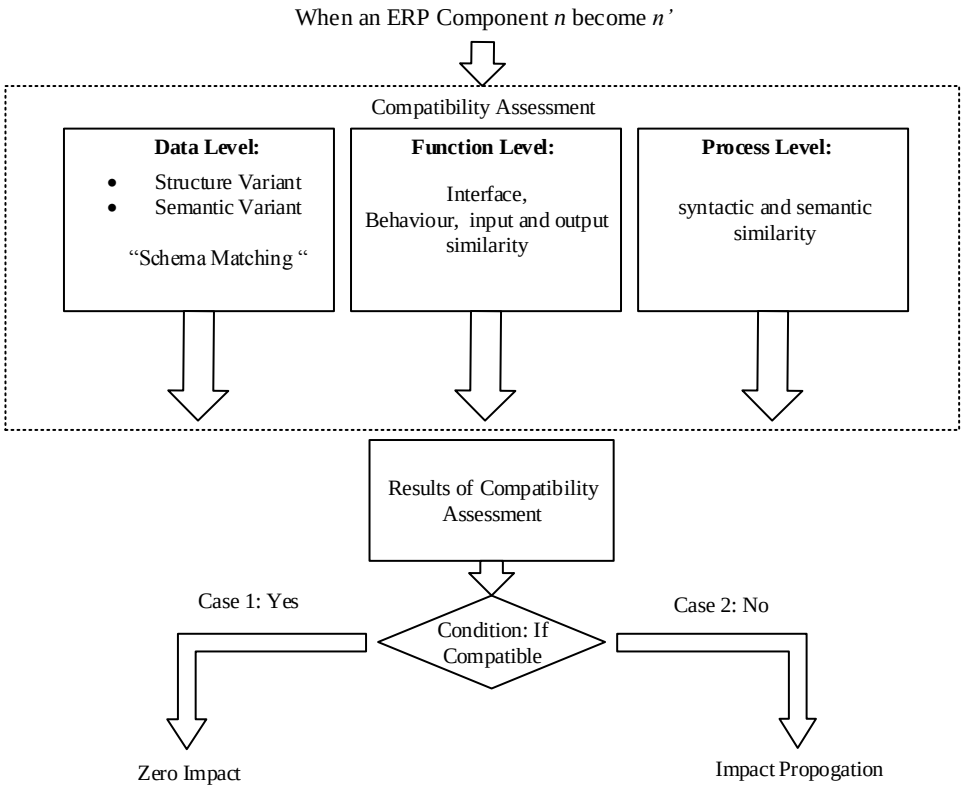


Figure 5-2 Compatibility assessments

As mentioned above, when upgrading an ERP component, such as a function, process or data object, from an old version n to a new version n', it is essential to assess the compatibility of the new version against the old one. This comparison requires a set of similarity metrics for measuring the compatibility value. The value of the metrics presents the compatibility degree, which determines whether the new component can substitute the old version, or not. The higher this value, the more likely the new component is compatible, while a low value means that the modified component is not compatible with existing ERP system (so, the impact of its implementation will have to be assessed by our impact mechanisms).

Let us revisit the example of Chapter 4, where the proposed change required updating a business function (i.e. creating purchase requisition) to provide additional information about the price range of purchasing a product or service, according to the history of the purchasing order. This change to the function requires retrieving additional information as an input from the purchasing history, in order to estimate a suitable price and display this

information in the purchase requisition document. Comparing this function with the old version, the new version has a different interface when compared to the old one, since it requires more data in input (i.e., the history of the purchase order) and gives more data as output (i.e., the suitable price estimate). Thus, the proposed change is not compatible with the existing ERP system, that is, the new function cannot "seamlessly" substitute the existing one. As such, the impact of this proposed change will have to be assessed.

Sometimes modifications of the ERP components can be considered as compatible with the ERP system configuration. The case is associated with the change in the format of the attribute of a data object from the string to an integer type that only requires an adjustment of the parameter settings in the configuration. Therefore, the modification performs with zero impact on the rest of ERP components since as a data value the integer type is a subset of the string type as a member therefore in this case the modification is compatible (e.g. a data value with string is also can be read by the same functionalities if it is changed from string type i.e. ABC12345 to an integer type, i.e.,1234.

In the remainder of this section, we review the requirements of compatibility for different levels of change in ERP systems, i.e. business data, functions and processes, to give the reader an understanding of those cases in which the impact analysis mechanisms described later in this section should or should not be run. The purpose of this investigation is not to provide a formal description of compatibility metrics, but rather and to provide the theoretical underpinnings of the notion of compatibility in different contexts.

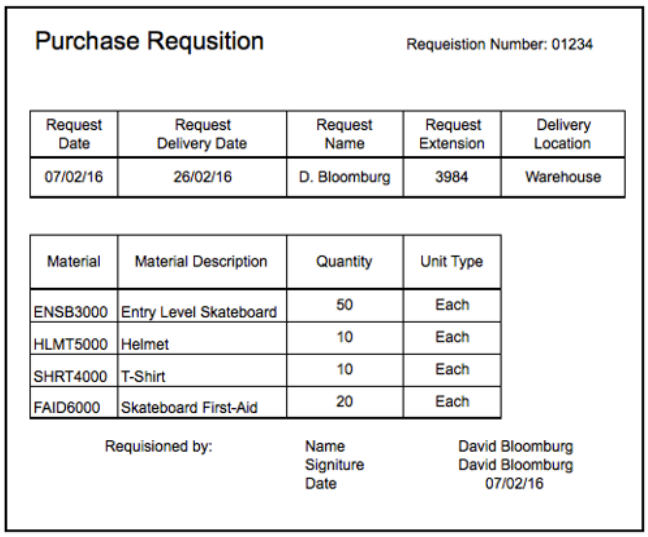
5.3.1 Compatibility Assessment Business Data

Change at the data level is not always guaranteed without a problem, such that the modifications of any attribute or data column can potentially interrupt the operation of a database system (i.e. all values stored in that column for all rows in that table are deleted). Data incompatibility exists when different and conflicting versions of the same data object appear in various places. From the literature survey, (Da Silva et al. 2007) compatibility at the data level is outlined in two forms, i.e. structure and semantic compatibility. The structure refers to the matching of the interface and the format of data, and the semantics refers to the behavioural change of the data model.

The compatibility evaluates the similarities of two data objects by analysing the common properties (Zhao and Zou 2011). We examine these features of data object through the utilisation of schema matching by describing similarity from work defined by (Kang and Naughton 2003). A schema is a set of related elements from a data object, such as tables, columns, and attributes. Data modification may refer to any change in data schema such that create, remove or rename tables, columns, indexes and foreign keys that represent as a data field in the data object.

Schema matching is the process of determining semantic correspondences or matches between two data object schema. A schema matching result or mapping consists of all possible matches between the elements of both schemes. First, one logical approach is to compare attribute names across the tables. Some of the attribute names will be clear candidates for matching, due to generic names or common parts of names (Bellahsene, Bonifati, and Rahm 2011). The second approach is to compare the attribute value in a case of attribute name mismatching.

Note that missing data might cause severe problems during run-time; e.g., an invoked function might crash or produce wrong outputs. Compared to the missing data, the problem of unnecessary data is less severe since it usually does not prevent a process model from being correctly executed. Nevertheless, unnecessary information in data objects should be avoided as they are not used anywhere in the system thus decrease model comprehensibility.



Purchase Requsition

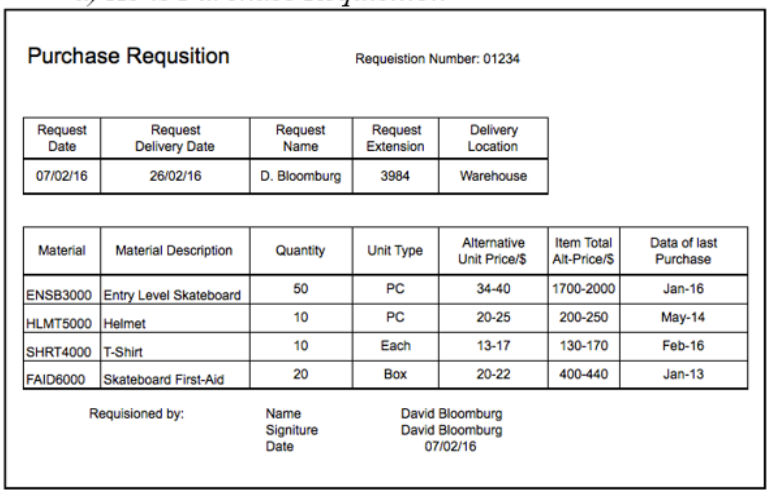
Requeistion Number: 01234

Request Date	Request Delivery Date	Request Name	Request Extension	Delivery Location
07/02/16	26/02/16	D. Bloomburg	3984	Warehouse

Material	Material Description	Quantity	Unit Type
ENSB3000	Entry Level Skateboard	50	Each
HLMT5000	Helmet	10	Each
SHRT4000	T-Shirt	10	Each
FAID6000	Skateboard First-Aid	20	Each

Requisioned by: Name David Bloomburg
Signiture David Bloomburg
Date 07/02/16

a) As-is Purchase Requisition



Purchase Requsition

Requeistion Number: 01234

Request Date	Request Delivery Date	Request Name	Request Extension	Delivery Location
07/02/16	26/02/16	D. Bloomburg	3984	Warehouse

Material	Material Description	Quantity	Unit Type	Alternative Unit Price/$	Item Total Alt-Price/$	Data of last Purchase
ENSB3000	Entry Level Skateboard	50	PC	34-40	1700-2000	Jan-16
HLMT5000	Helmet	10	PC	20-25	200-250	May-14
SHRT4000	T-Shirt	10	Each	13-17	130-170	Feb-16
FAID6000	Skateboard First-Aid	20	Box	20-22	400-440	Jan-13

Requisioned by: Name David Bloomburg
Signiture David Bloomburg
Date 07/02/16

b) To-be Purchase Requisition

Figure 5-3 Example of updating purchase requisition

The example in Figure 5-3 demonstrates two data objects for purchase requisition document. Figure 5-3-a) presents, the current version of the purchase request in the ERP system while Figure 5-3-b) extends this document by upgrading the current version with extra information. The compatibility between the two types of data objects begins with the similarity analysis of the attributes (label), attribute values and the relationship with other data (foreign key). The procedure for the comparison is as follows:

- First labels for both data objects are compared, including the table column name.

- Second, check the similarity of the data objects attributes. The data object in Figure 5-3-b) has three extra attributes (i.e. Alternative Unit Price, Total Item Price and Date of last purchase) that capture this information through the new association with another data object (i.e. purchase info record known a repository of purchase history).
- Third, check the similarity of the attribute values. The attribute value of the unit type for new purchase requisition shows more option compares to the previous one by extending the enumeration type of attribute value.

From the above comparison confirms that the new data object in the Figure 5-3-b is not compatible with the existing one (i.e. as-is purchase requisition). Consequently, it is necessary to continue the modification assessment by running impact analysis to define the ERP components that are associated with this data object.

5.3.2 Compatibility Assessment Business Function

To assess the compatibility of business functions, we use the same concept as used for service compatibility. Three levels of compatibility for service modification have been previously reported by (Pianwattanaphon and Senivongse 2007), i.e. interface, behaviour and input and output (data type) compatibility. The same levels are considered by (Yan, Dijkman, and Grefen 2010) in the context of compatibility for business functions

The first type of similarity (function interface) considers a set of common properties of functions such as the function name, and function description (Wu and Wu 2005). Properties of this category describes common information of function in string. So to measure the similarity of string, WordNet measure is employed to calculate the minimal distance between two words (Wu and Wu 2005). The second similarity metric (behaviour) concerns the special properties that contain a set of unique properties of the function like the attributes of functionality. For instance, the attributes of function can be the module and business processes that the function has associated with them. The similarity of these properties can be measured through the calculation of the attributes of the function. Finally, the third type of similarity (input and output) refers to the input and outputs of the operation(s) of a function (Wu and Wu 2005). The interface of the function contains a set of

operations, and each operation has various input and output parameter. For these parameters, the similarity is measured based on the parameter name and parameter data type. Therefore, interface similarity assessment relies on the comparison of the operation input and output parameters (Wu and Wu 2005). This interface similarity checks for function f that is replaced by function f' as follows:

- For each operation in business function f, there is an operation in *f'* with the same name.
- There is at least one input for each operation that taken place in *f,* the input should be the same as the operation in function *f'*.
- The output for each operation in function *f'* should be the same as output for the operation of function *f'*.

The semantic aspects are related to the goal of the function and correspond to the name used for the whole function, the operations, and the input and outputs, whereas the syntactic aspect can state the compliance between the input and output structure and adapted data types.

Let us consider the case of upgrading a function (see Figure 5-4). The change is applied to the existing function (i.e. create purchase order) by adding an extra feature to include the tax when calculating the total cost of the purchase.

Clearly, this function requires no further data object to associate with it, besides there is no further change impact concerning a new attribute to the data object (i.e. purchase order) and the business process can continue without any errors during the execution. Therefore, this proposed change is compatible with the existing system and only requires a minimal adjustment in the configuration settings to encompass the tax during the addition of all purchase prices. The above example satisfies the compatibility regarding the following similarity assessment:

- Based on the interface similarity: the properties of the function remain the same (i.e., function name and function description)
- Based on the Behaviour similarity: the function still in operates in the same business process and same module

- Based on the input and output similarity: no extra data are required as input and no additional data create as an output and the operations are the same for both functions.

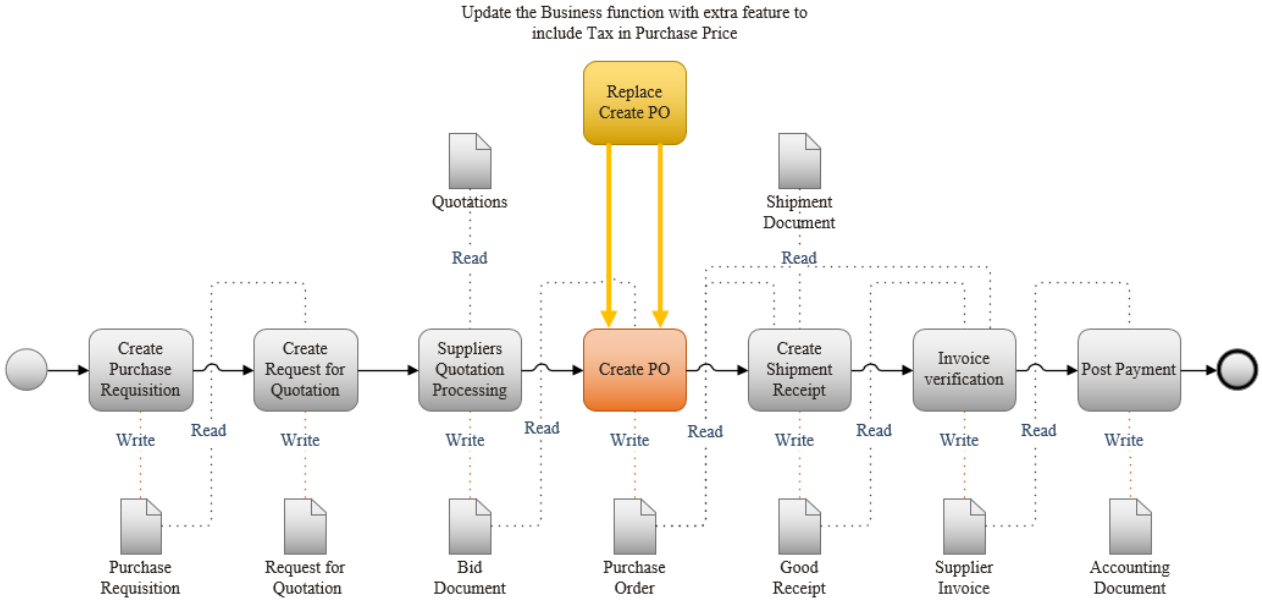


Figure 5-4 Example of Updating Function (Create Purchase Order)

5.3.3 Compatibility Check Business Process

According to the (Dijkman, Dumas, and García-Bañuelos 2009), research that identifies the notion of compatibility for business process level based on the representation of the process model. Such that compatibility is used for process model components in EPC, BPMN or Petri Net Process model. Different compatibility types are presented as a measure to check the degree of similarity between two process components in ERP system environments. This similarity check shows how these compatibility measures can be verified. In the context of similarity, we define three fundamental similarity metrics.

- Label similarity which is based on a comparison of the labels that appear in the two process models using syntactic and semantic metrics i.e. function label, process label.
- Structural similarity is measured based on graph representation matching, possibly taking into account text similarity as well.

- Behavioural similarity is measured according to the execution semantics of process models.

According to (Dijkman et al. 2011), the first compatibility mechanism exploits the fact that process models are composed of labelled nodes. These metrics start by calculating an optimal matching between the nodes in the process models by comparing their labels. Based on this matching, a similarity score is computed taking into account the overall size of the models. The second class of metrics is structural. It is built on the observation that includes nodes in process models with their relations through a mathematical graph. This technique is used for graph comparison based on graph edit distance, which is commonly used in information retrieval. The third class of metrics is Behavioural, in the sense that it take into account the causal relations between tasks in a process model. These causal relations are presented in the form of a causal footprint, which provides an abstract representation of the behaviour of a business process model.

Besides the work by (Dijkman, Dumas, and García-Bañuelos 2009) addressed the concept of causality graph or causal footprint as a set of activities and conditions that are used to check the behaviour of a business process. A causality graph represents behaviour between a set of activities using two relationships, namely look-back and look-ahead links. For instance a look-ahead link from activity to a (non-empty) set of activities, we say that the execution that activity leads to the execution of at least one of the activities in the set. Furthermore, for a look-back link from a (non-empty) set of activities to an activity, we say that the execution of the activity is preceded by the execution of at least one of the activities in the set.

Figure 4-5 shows an example of business process similarity. It constitutes a simple process of a product query based on BPMN notation. The similarity techniques should return the value that indicates the extent to which these two processes are similar to each other.

In order to assess the similarity first, we start by comparing the labels for each node in the process model (i.e. functions). As such the assessment shows there are some differences between the functions, for instance, buying goods in process 1 and buy special goods online in process 2 and the same scenario for the functions Receive goods and goods receipt. The similarity measures the distance between the two labels for each function.

Therefore, it is critical to consider the similarity of these functions to ensure they provide the same result as the functions in the process 1. The additional assessment concerning similarity at the process level is the structural representation of the two business processes. This considers the graphical representation of each process that includes the comparison of the process features (i.e. start, end, sequence, split, joint) for each function.

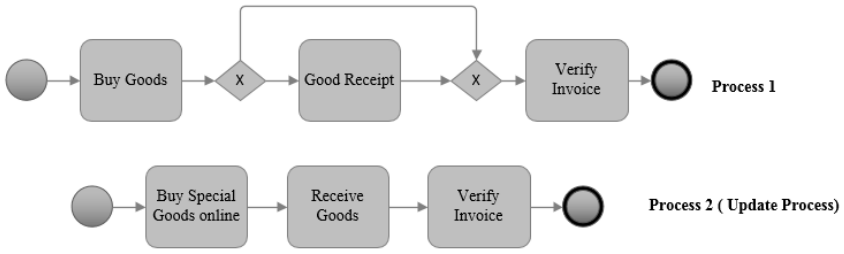


Figure 5-5 Example of Updating Business Process

Suppose that the two functions of goods receipt and receive goods both produce the same result, based on the structural representation one has the sequence relation and the other has the join which makes these two process to become less similar to each other. So the above case shows the result of incompatibility when upgrading the business process; therefore, it is important during the change process to understand the impact of modification in the ERP system.

5.4 Critical Point for Running Instances Affected by Change

This section describes some key aspects of the impact analysis assessment related to how to manage running instances upon the modification of any ERP components. First, this section explains the migration policies for managing the change during the runtime. Then, demonstrates a technique referred to as the critical point on how to apply migration policies for those instances affected by a proposed change as identified by our impact analysis.

5.4.1 Migration Policies for Running Instances Affected by Change

The impact of modification at the design time allows the developer to reconfigure, or redesign the components while the runtime refers to the instances, which require an action for managing and controlling the modification more efficiently. Note that in this thesis when we refer to the impact of instances of any process, function or data, there is always a case

where the effect of a modification results in changing the business process model of the ERP system as well. For instance, updating the purchase requisition also impacts all the running instances of the purchase process.

There are two fundamental principles apply to manage the running instance during the migration of the modified component.

1. The first principle implies on anticipating change and migrating the instance where the instances are in a safe state. This means that it is not safe to migrate instance to the new process for which the process or data object that is changing still has to be executed. A safe state is when the change can be applied without causing any errors during the remainder of the execution of the process instance.
2. The next principle is referred to remove the old version of the process as soon as all process instances are transferred in the state where the execution can continue based on the modified component. In this context, techniques are introduced for dealing with running process instances and the policies to the migration of instances, without violating any correctness and soundness properties.

The work from (Casati et al. 1998) defines four types of migration policies during runtime to manage running instances, namely, as flush, abort, migrate, and ad-hoc:

- Flush: This strategy means that the current instances will continue execution until completed using the old process model.
- Abort: All the active instances are terminated when the process model is changed, and new instances will start according to the new process model. This is advisable when there are few instances running in the system and mostly at the initial stage of execution. This policy only applicable to the case when modification is related to the delete operation;
- Migration: The running instances are migrated to the new process model based on certain conditions that explain in the next section (e.g. when the instance is passed the critical point).
- Ad-hoc: This strategy is only recognised as an unforeseen exception, for which the modification only applies to some running instances; however, the main business process model may remain as unchanged.

Consider the purchasing process and assume that due to newly emerging requirements, a modification is requested. Suppose this process has numerous instances of business transactions, which are still active in the system. Therefore, applying migration policies manually to each active instances of this process is hardly a realistic option. Similarly, stopping all the active process instances, aborting them or restarting them could not be a viable option. Therefore, different actions can be made to handle on applying migration policies during the execution of process instances. This requires a technique to assign migration of instances during the change process. We exploit the work from (Casati et al. 1998) into our impact analysis that suggests one of the above strategies that are more appropriate during the implementation of the change according to the execution state and the number of instances of a given process.

The next section explains the notion of critical point to define a migration rule for each impact instances in a controlled and correct manner. A fundamental challenge for the ERP system is the ability to respond adequately to process change when a thousand transactions are running in the system. The easiest way to correctly complete these running process instances is to continue their execution based on the original process model until reaching a certain point in a process model that the modification has no impact on the process instances.

5.4.2 Critical Point in Running Instances Affected by Change

The notion of critical point in a running instance affected by change fulfils the requirement to take a decision on when to migrate the process instance from the old version *p* to the new version *p'*.

The critical point is described as a specific point in the process model. A running instance can be migrated to the new process model if its execution has passed this point. In a nutshell, let us consider a simple process P constituted by 5 activities A, B, C, D, E, as shown in Figure 5-6 and let us assume that the proposed change occurs in C (the change may refer to introducing a new version of function C (i.e. C'), a non-compatible modification of the input/output data object of C, or a process change that does not involve D or E). The function C will be the critical point of all the running instances of process P. Instances that

have passed C in their execution will be safe, that is, they can continue their execution according to the old or new process model, since the part of the process still to be executed (D and E) is not affected by the proposed change. Instances that have not passed C are not safe to be migrated to the new process model, because they still have to go through the last point (activity C) where the proposed change occurs.

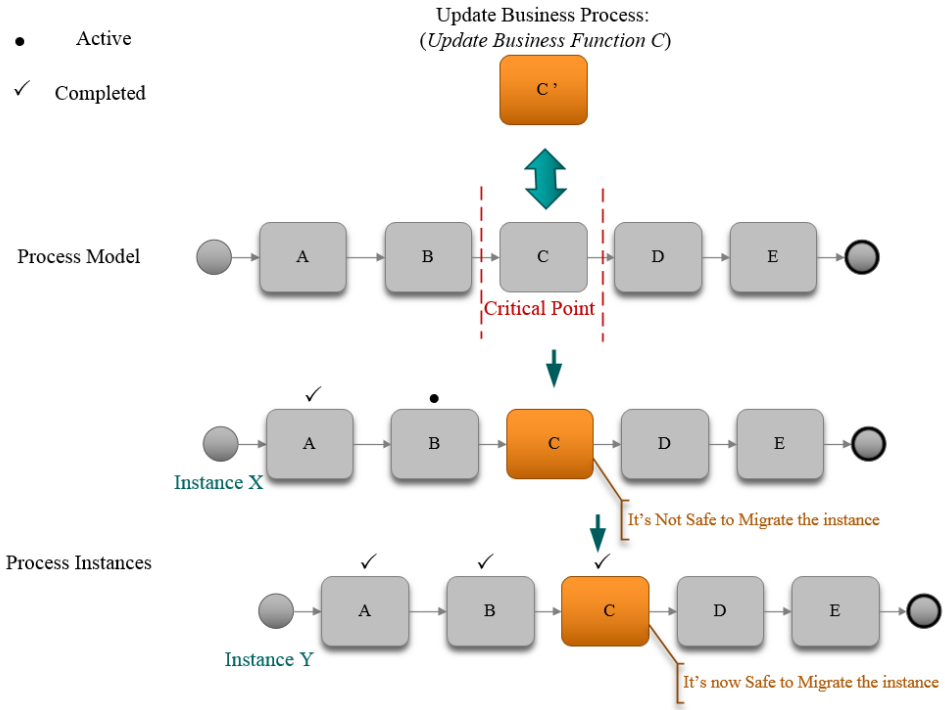


Figure 5-6 Example of Critical Point

Based on the notion of critical point, we can decide whether a particular change is correctly applicable to a process instance in the current state of analysis or not. Thus, it is important in order to identify the stage of execution of each process instance to be able to apply migration policies. Through the identification of the function instances state, the impact analysis can verify the transition states of the process instance. When instances pass the modification point, then it is appropriate to migrate process instance without any difficulty to continue the execution. Based on the state of the execution at a critical point, different actions can be made to handle the changes.

The calculation of critical point is not as simple as in the case of the example of Figure 4-6. Our impact analysis considers two cases of how to compute the critical point of business process instances to apply the migration:

- Simple Case: refers to the situation when there is no integration of the process affected by the change with other business processes.
- Complex Case: refers to the more complex situation when the process affected by the change is integrated with other business processes in the form of sub-process that the migration policies are treated differently for process instances that include the execution of another process instance as a result of integration.

For each of the above cases, the critical point is defined differently. First, we discuss on how to assess the migration policies for the business process without integration as a Simple case. The critical point plays a significant role in our impact analysis; in particular, it defines the procedures for active instance in the ERP system. As soon as this determination of critical points achieved, the proposed action can be made. This provides an efficient way for identifying the migration rule for process instances. Note that the aim of applying the migration policies to the instances is not used for changing the instances based on modification but also to define the impact for handling the instances during the change process. This is one aspect of the modification process for change monitoring.

The case in Figure 5-7 illustrates the process model at design time with three examples of the process instance. Consider the scenario is shown in Figure 5-7 the change is applied to process *P1* and results in process *P1'*, in which the function *A2* is removed from the process model. So to handle the running instance the correctness notion is necessary for deciding when the process instances can migrate from *P1* to new process *P1*'. Thus, to transfer the active process instances to the new process model, the state of process instances is required to pass the point (i.e. in the above example is function *A2*) where the modification of a process model is relevant to the active process instance. This point is known as the critical point where the transition can apply to the process instance from *P1* to *P1*'.

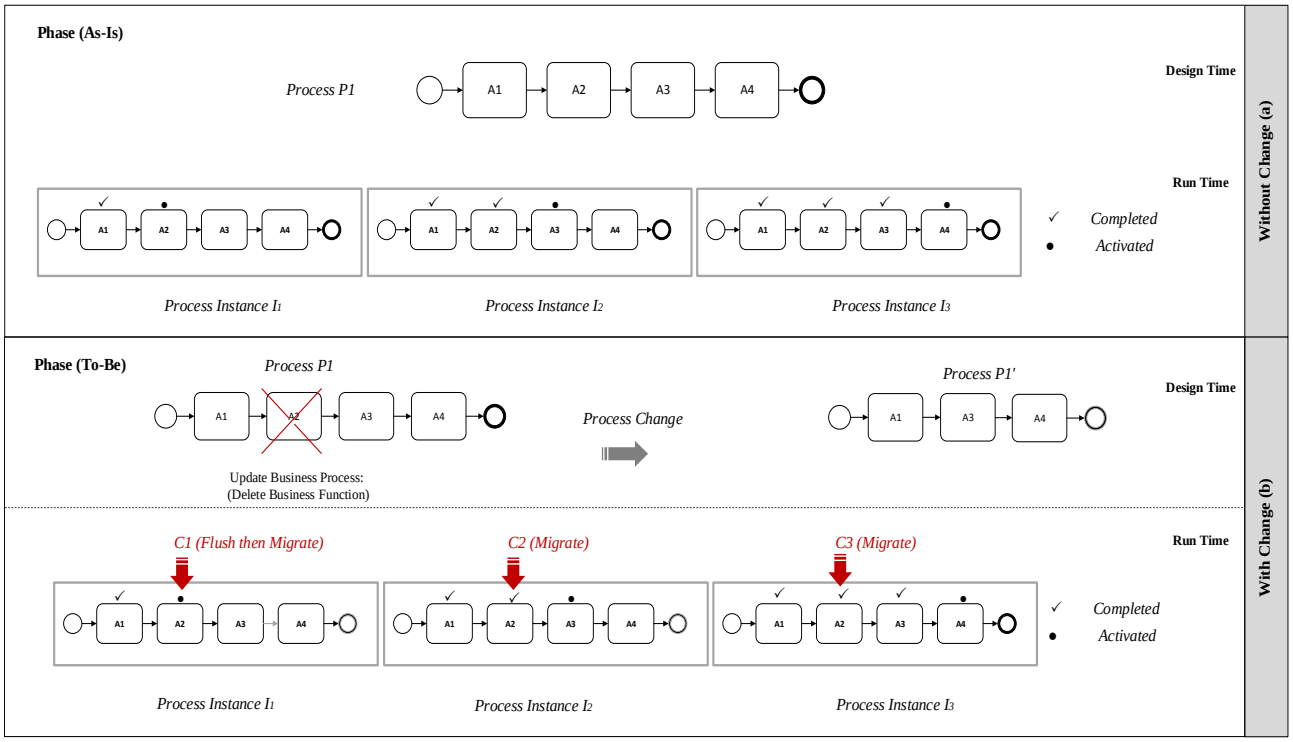


Figure 5-7 Notion of Critical Point for Simple Case

In the above example, the critical point is defined when the process instances have completed the execution of function *A2*, and then all the active instances can migrate and continue the execution according to the new process *P1'*. According to section 5.4.1 that classifies the migration policies, when the critical point is in the state inactive or active the flush policies apply until passing this point. Flush policies for the above example mean letting the process instances proceed according to the initial specification before passing the critical point. Once the state of the critical point transfers to complete, then it is possible to transmit the process instance according to the modified version.

As for this case, when the critical point defines as completed the migration policies can apply to the instance. We now discuss the notion of critical point in the complex case (see Figure 5-8).

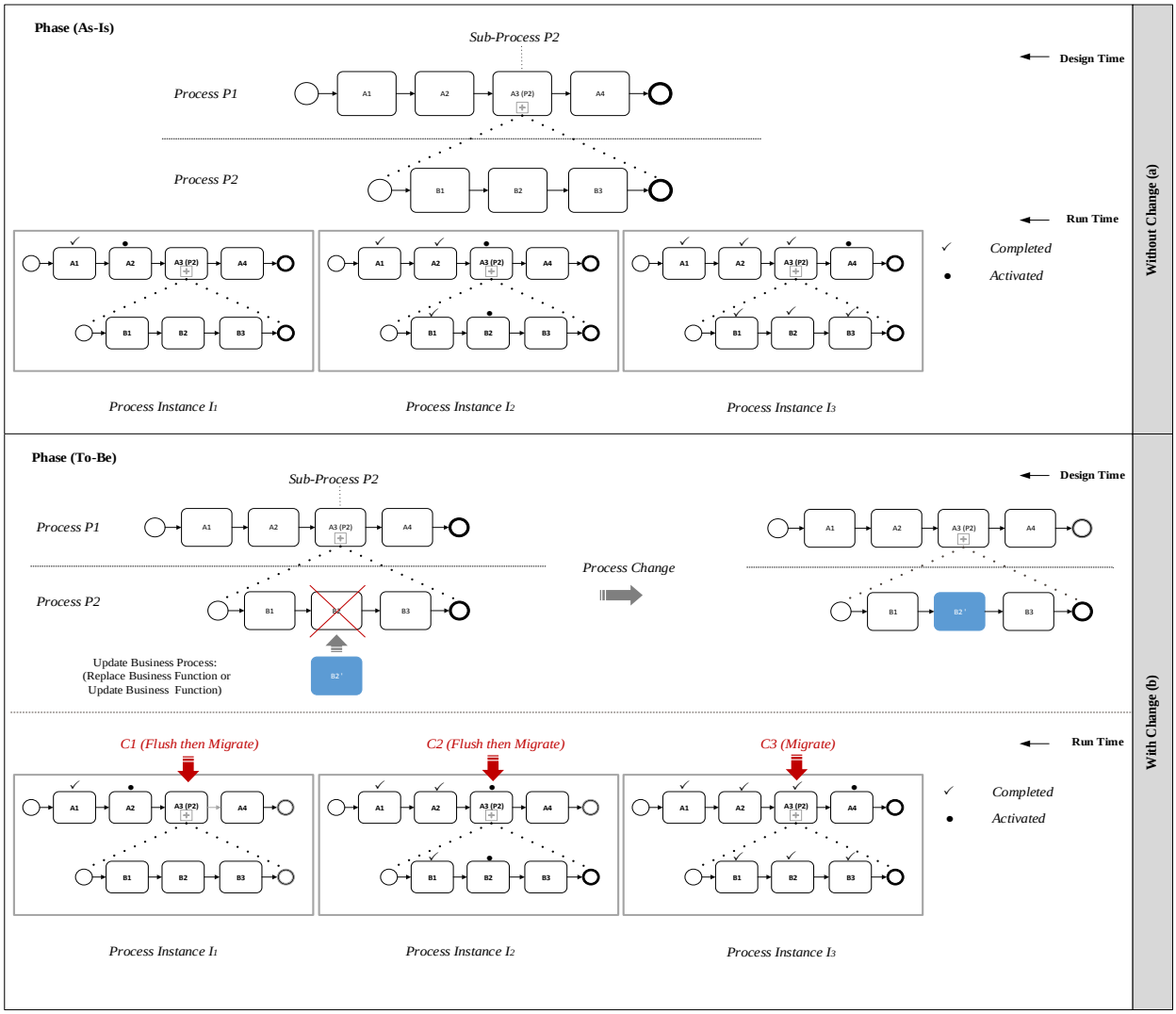


Figure 5-8 Notion of Critical Point for Complex Case

As far as integration concerns, the notion of a critical point is specified differently compare to the above scenario. Figure 5-8 distinguishes the analysis of the critical point for process instances with integration. The example shows the process *P2* is integrated with process *P1* as a subprocess. Suppose the modification applies to the process *P2* by replacing the function with another that concludes as updating the process *P2*. When the execution of process *P1* is started presumably, the instance of *P1* creates the instance for the process *P2* as well as a result of integration. Therefore, to evaluate and apply the migration policies it is essential to identify the critical point for both business processes *P2* and *P1*. However, there

is a different method we conduct for the identification of the critical point for each business processes to apply migration policies.

Technically process *P2* is considered as a business function of process *P1* by which any modification in *P2* considers as an update the function block in process *P1*. In order to apply the migration policies for instances of process P2 the critical point considers as the point where *P2* is integrated with process P1. This is entirely different compared to the previous case.

The critical point in the previous case defined as the point where modification is applied in the process model (i.e. process *P2*) whereas in this case to define where the process is called or is in used as a subprocess of another business process (e. Process *P1*). The migration performs when all instance of process *P1* are completed or already passed at the point where process *P2* is integrated. Otherwise, the Flush policies are carried out to continue the execution until the instance passes this point and then migrate.

This also involves all the instances of process *P2*. Therefore, the migration policies of process *P2* depend on the state of the process *P1* and how far is the instance away from completion. The example of the above case is when there is a change in the purchasing process, and the process is integrated with the manufacturing process to supply the raw material. In practice, it is not feasible to migrate the process instance until the process instances are considered in the state where the modification is not violating the execution. For instance, in ERP system first create a production order to make a product that includes purchasing of the raw material during the manufacturing process. So any changes in purchasing also impact the instance of production. Therefore, it is important to check first the state of the manufacturing process and then apply migration for those instances that already passed the purchasing.

The next section applies all concepts defined thus far in this chapter in the specification of the impact analysis mechanisms.

5.5 Propagation Mechanism

In Section 4.3 we have reviewed different types of change in ERP system, however, all rely on the analysis of dependency relations for estimating the propagation of changes.

Consequently, to specify the algorithms, we first need a set-theoretic representation of ERP components and their dependencies (see Table 5-1). This is based on ERP dependency meta-model of Section 4.2.

Table 5-1 Set theoretic representation of ERP components

Set	Notation	Predicates
ERP system	$ERP = < M, PRO, F, D >$	
Module	$M = \{m_n\}_{n=1,2,\ldots,N}$	
Business Data	$D = \{d_k\}_{k=1,2,\ldots,K}$	
Business Function	$F = \{f_i\}_{i=1,2,\ldots,I}$	$Interact\ (f_i, d_k, type)$, $type\{I, O\}$ *input and output* $Include\ (m_n, f_i)$ Module include functions
Business Processes	$PRO = \{pro_p\}_{p=1,2,\ldots,P}$	$Uses\ (pro_p, f_i)$ $Call\ (pro_a, pro_b)\ , a \neq b$ $Called\ by\ (pro_a, pro_b)\ , a \neq b$
Process instance	$INS_p = \{ins_{p,j}\}_{j=1,2,\ldots,J}$	$Create(pro_p, ins_{p,j})$
Function Instance	$FINS_i = \{ins_{i,q}\}_{q=1,2,\ldots,Q}$	$Create(ins_{p,j}, ins_{i,q})$
Data Instance	$DINS_k = \{ins_{k,z}\}_{z=1,2,\ldots,Z}$	$Create(ins_{i,q}, ins_{k,z})$

The set-theoretic representation is required to give an explicit specification of the algorithms for the preservation of the change analysis of the ERP system (Comuzzi, Vonk, and Grefen 2012). Classes in the meta-model in Chapter 4 Figure 4-5 are represented here as sets, e.g. business data or business processes, whereas predicates capture the relations between the classes. Predicates take as arguments elements of the corresponding sets (Comuzzi, Vonk, and Grefen 2012). Predicates can be thought of as an operator or function that may return a value either true or false.

Table 5-1 specifies a list of set elements of an ERP system with the set notion and related predicates. The predicate in this table defines as the statement that shows the relations with the other member of a set, which has to be true. Table 5-1 explains the following definitions:

- ERP system is represented as a tuple of <*M, Pro, F, D*> where M is the set of modules, *Pro* the set of processes, *F* the set of business functions, and *D* the set of data objects currently in the ERP system
- There are N modules m_n in the ERP system and K data objects d_k.
- There are **I** functions f_i in the ERP system. The predicate $Interact\ (f_i, d_k, type)$ signifies that the function f_i uses the data object d_k as either input (type= *I*) or output (type= *O)*. The predicate $Include\ (m_n, f_i)$ implies that the function f_i is part of the module m_n.
- There are *P* processes pro_p in the ERP system. Each process model uses function from set *F*. The predicate $Uses\ (pro_p, f_i)$ indicates that the process pro_p uses function f_i, and the predicate $Called\ by\ (pro_a, pro_b)$ specifies that the process pro_a is called by the process pro_b as a sub-process. The predicate $Call\ (pro_a, pro_b)$, indicate that the process pro_a calls as a sub-process the process pro_b
- There are J Process instances $ins_{p,j}$ of process pro_p in the ERP system. The predicate $Create\ (pro_p, ins_{p,j})$ signifies that the process pro_p creates a new process instance $ins_{p,j}$. Each process instance of INS_p representing different case.
- There are Q Function instance $ins_{i,q}$ that belongs to each process instances in ERP system. The predicate $Create(ins_{p,j}, ins_{i,q})$ signifies that the process instance $ins_{p,j}$ during the execution create an instance of function $ins_{i,q}$.
- There are Z data instance $ins_{k,z}$ that belongs to each process instances in ERP system. The predicate $Create(ins_{i,q}, ins_{k,z})$ signifies that the function instance $ins_{i,q}$ during the execution create an instance of data $ins_{k,z}$.

One important aspect of impact analysis is how to specify a change that could be understood by the algorithms (Lee 1998). After defining the ERP system components and dependency relations of ERP entities in the form of set theory, then it is required to establish criteria to deploy change mechanism through the set of algorithms. As explained in Section 3.4 when business analysts want to assess the proposed modification, they need to analysis the end-user requirement and define which parts of the ERP system are going to be affected through some criteria (Lee 1998). The following listing summarises the criteria for classifying change operations that cover in our taxonomy in section 4.4:

- Composition Type: Atomic, Composite.
- Type of Operation: Add, Delete, Update
- Level/Scope of Change: Function, Data Object, and Process

After the change criteria have been defined, our algorithms start computing the change impacts. The algorithms will find all members in the existing ERP system that could be impacted. According to the specified change criteria, the algorithms first assess the compatibility at each change level (i.e., data, function, and process) then based on the characteristics change level, inheritance, and dependency relationship with other components of ERP system computes the impacted items in the system.

5.5.1 Impact Analysis Algorithm: Update Algorithm

Based on the change taxonomy defined in Chapter 4, our framework defines an impact analysis algorithm for each type of change, i.e., add/delete/update of data object/function/process. In this section, we focus on the impact analysis algorithm for the "Update Data Object" type of change. This is the most complex type of change to handle as changing a data object may bear an impact on all the functions that are using it and all the business processes that are using such functions (and the related runtime instances). As such, the algorithm involving a change at the data object level is discussed in this chapter as exemplar cases. The other algorithms are reported in Appendix A.

5.5.1.1 Business Data

To facilitate the reader, we provide a high-level representation of the algorithm using natural language-based constructs (Algorithm 1) and, after having explained the algorithm, we also provide detailed representation exploiting the set-theoretic representation given before (Algorithm 2).

Algorithm 1: Updating Business Data:

```
1: Input: Update Data object d_k to d_k'
2: begin
3:     Run Compatibility Check for the Data objects (d_k, d_k');
4:     if Data object d_k Compatible with d_k' then
5:       Implement New Data object d_k' ;
6:       Retrieve all Functions using d_k;
7:       Retrieve all Process using Functions Found;
8:       for each Process Found do;
9:           Retrieve all Process Instances;
10:          Migrate all Process Instances to New Data Object d_k';
11:      end for
12:      Delete Data object d_k;
13:    else
14:      Implement New Data object d_k' ;
15:      Retrieve all Functions using d_k;
16:      Retrieve all Processes using Functions Found;
17:      for each Process Found do;
18:          Check Integration with other Processes;
19:          for each Integrated Processes Found;
20:              Run INTG_PRO;
21:          end for
22:          Retrieve all Process Instances;
23:          Create set for Process Instances without Process integration;
24:          Create set for Process Instances with Process integration;
25:          for each set do;
26:              Compute the critical point;
27:              if Process Instances Passed the Critical Point then;
28:                  Migrate all Process Instances to d_k';
29:              else
30:                  Continue the execution for all instances till passes the Critical Point
31:                  Migrate Instances to New Data Object d_k';
32:              end if
33:          end for
34:      end for
35:      Delete Data object d_k;
36:    end if
37: end
```

Algorithm 1 is triggered when a business analyst attempts to assess the modification of existing data object in the system such as adding an extra field or changing the attribute type. Line 1 begins with the specification of the modification (i.e., change level and change

type) that in this algorithm refers to updating the existing data object by replacing the old version d_k with the new data object d_k' in the system.

As explained before it is always required to check the compatibility in a case of replacing the existing entity with a newer version. Line 3 then checks the compatibility of the new data object d_k' against the previous version d_k (see Section 5.3.1). If the two data objects are compatible, then the impact analysis concludes as the change has zero effect on the existing ERP system and the new data object can simply substitute the old one. For the run-time, all the instances affected by the change can be migrated to new process models involving the new data object. Therefore, Line 6 and Line 7 starts with finding all the functions and business processes in which the current data object d_k is used. The main reason for this assessment is to maintain the consistency throughout the system, in particular for those instances that are still active in the system and using the old version of the data object. Once the migration is completed for all active instances, the old data object d_k needs to be removed to eliminate the creation of the same data object twice in the system.

The other case is the one in which the two data objects d_k and d_k' are not compatible. In this case, it is essential to compute the ERP components that are affected by the change based on the dependency relationships identified by the ERP dependency meta-model in our framework. From line 13, the actual impact propagation rule applies, that potentially returns a list of impact items. First, the algorithm starts retrieving all the business functions that the data object d_k is created as the output or is in used as the input. Then retrieves all the business processes where the business functions found from the previous step. Line 17 checks process integration for all business process obtained from the previous section. If there is any integration with other business processes, then impact analysis calls the algorithm $\mathbf{INTG_{PRO}}$ for further checking the integration of process found and to monitor and control the running instances that result from process integration. The process will be completed when the algorithm $\mathbf{INTG_{PRO}}$ is finished finding all possible process integration and instances. Then at line 22 the algorithm represents the migration policies by retrieving

the process instance of all process found at line 16. Since some of the process instances are created as a result of process integration, then it is essential to split the process instances into two sets. One set refers to the process instances in integration, whereas the other is used for the case without integration.

The reason for defining two sets is explained by the notion of critical point that discussed in the previous Section 5.4.2. Based on this definition, the critical point is specified for both cases accordingly. Then for each process instance if the execution passes the critical point then the instance can migrate to the new process model. While if the process instance has not passed the critical point, it is recommended by impact analysis to continue the execution until the state is transferred to complete at the critical point, then the instance can migrate to the new process model. Once all processes instances are completed the assessment, then impact analysis allows to remove the old version of the data object from the system, as it is not advised to be used during the execution of a new transaction in the system.

Algorithm 2: Updating Business Data:

Input: UpdateData (d_k, d_k')
begin
  COMPATIBLE ← CompatibilityCheck (d_k, d_k')
  if COMPATIBLE = TRUE **then**
    CREATE d_k' //Change can be implemented
    FUNC ⊆ F ← $\{f_i: Interact\ (f_i, d_k)\}$
    for each f_i ∈ FUNC **Do**
      PROC ⊆ PRO ← $\{pro_p: Uses\ (pro_p, f_i)\}$
      for each pro_p ∈ PROC **Do**
        INST ⊆ INS ← $\{ins_{p,j}: Create\ (pro_p, ins_{p,j})\}$
        Migrate all ins ∈ INST_{Pro} to d_k'
        Delete d_k
      end for
    end for
  else
    CREATE d_k' // COMPATIBLE = FALSE
    FUNC ⊆ F ← $\{f_i: Interact\ (f_i, d_k)\}$
    for each f_i ∈ FUNC **do**
      PROC ⊆ PRO ← $\{pro_p: Uses\ (pro_p, f_i)\}$
      for each pro_p ∈ PROC **do**
        INTG_{PRO} ⊆ PRO ← $\{pro_p: Called\ by\ (pro_p, pro_q)\}$
        for each pro ∈ INTG_{PRO} **do**
          Call $\boldsymbol{INTG_{PRO}}$
        end for
        $\text{INS} \subseteq \text{INS}_p \leftarrow \{ins: Create\ (pro_p, ins)\}$
        $\text{INS}_{with\ INTG} \subseteq \text{INS} \leftarrow \{ins: Create\ (ins_{p,b}, ins_{q,a})\}$
        for each ins ∈ INS **do**
          if ins ∈ $\text{INS}_{with\ INTG}$ **then**
            CP ← calculateCriticalPoint (pro_p, pro_q)
          else
            CP ← calculateCriticalPoint (d_k, pro_p)
          end if
          PASSED_CP ← checkCriticalPoint (ins, CP)
          if PASSED_CP **then**
            Migrate
          else
            Flush then Migrate
          end if
        end for
      end for
    end for
    Delete d_k
  end if
end

Algorithm 2 provides the detailed representation exploiting the set-theoretic that explains the same concept and mechanism as algorithm 1 in a more formal description.

To have a better understanding of the above algorithms let us revisit the simple example explained in Section 5.3 of a change scenario for updating the business data (i.e.

purchase requisition). In this example, as illustrated in figure 5-9 the compatibility assessment first defines the modifications as incompatible, according to the assessment explained in section 5.3. Therefore, the algorithm starts analysing the impact items, which cover the second part of the algorithm for updating the business data.

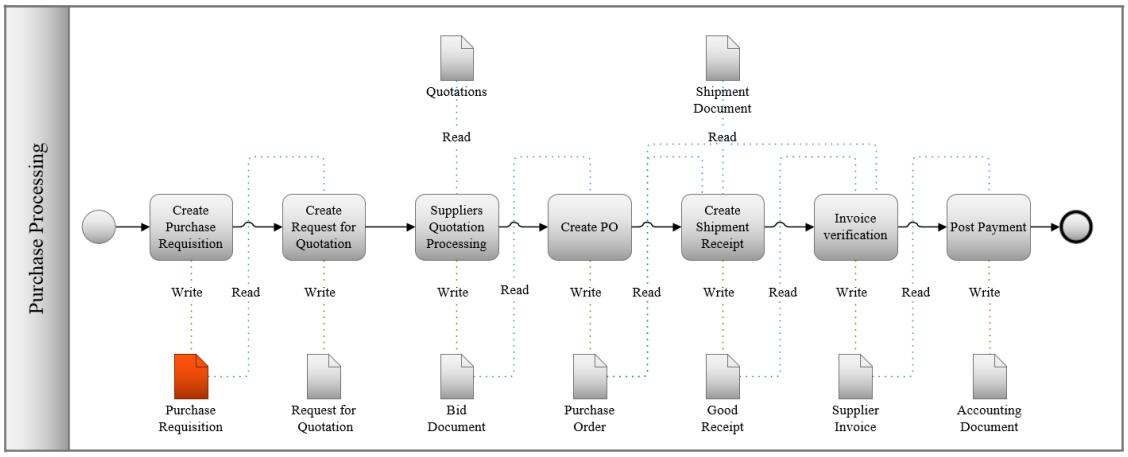


Figure 5-9 Example of Update Purchase Requisition

The impact mechanism first finds the functions that this particular data object has been used during the execution in the form of input and output (i.e. Create Purchase Requisition and Create Request for Quotation). Then the algorithm retrieves the business processes, which the above functions are used during the execution (i.e. purchase process). Suppose that the business process has some active business transactions running in the system that results from of various purchase requests of different products. Therefore, the impact analysis has to capture these transactions to manage the migration of modification appropriately.

As explained in section 5.4 the transactions of the business process can be migrated to the new process model if the execution has passed the critical point. In this case, the function (Create Request for Quotation) will be the critical point of all the transactions of the purchasing process. Transactions that already passed the function (Create Request for Quotation) will be safe; this means that they can continue their execution according to the old or new process model since the other part of the process is not affected by changing the purchase requisition document.

For those transactions that have not passed the function (Create Request for Quotation) are not safe to be migrated to the new process model because they still using this data object during the execution.

Therefore, the impact analysis suggests continuing the execution until the transaction passes this point in order to migrate the process according to the new process model. As soon as the migration for all transactions is finished, the impact analysis allows deleting the older version of the purchase requisition document so that the system can only perform based on the new version of a data object.

5.5.1.2 Process Integration

Algorithm 3 $\boldsymbol{INTG}_{pro}$ is triggered when there is process integration found during the analysis of another business process. Line 1 begins with checking process integration, which first checks and retrieves all the business processes calls business processpro_a. If there is no process found, then the algorithm proceeds by retrieving all the process instances of pro_a that are created during the execution. Same as the previous algorithm the instances are divided into two sets where there is integration with other business process and without integration. Then for each set defines the critical point. According to the notion of the critical point in section 5.4.2 for each process instance impact analysis specifies the execution state of instances, such that if the instance is in the active or waiting (i.e. inactive) state then it is recommended to continue the execution until the execution passes the critical point. It is reasonable to migrate the modification within the business process to the new version without any interruption on running instances.

However, if the execution has been passed the critical point, then the migration can be performed without any problems. The above scenario is developed for the case where there is no further integration of a business process. In line 5 explains the scenario when the algorithm found more integration to the business process. Therefore, the analysis continues as recursive functionality until there is no further process integration found in the system (i.e. called by other business processes).

Algorithm 3: $INTG_{PRO}$

```
1: Input: Check Process Integration of pro_a
2: begin
3:     Retrieve all Process calling Process pro_a;
4:     for each Process Found do;
5:         Call INTG_PRO
6:     end for
7:     Retrieve all Process Instances pro_a;
8:     Create set for Process Instances without Process integration;
9:     Create set for Process Instances with Process integration;
10:    for each set do;
11:        Compute the Critical Point;
12:        if Process Instances Passed the Critical Point then;
13:            Migrate all Process Instances
14:        else
15:            Continue the execution for all instances till passes the Critical Point
16:            Migrate Instances to new the modifciation
17:        end if
18:    end for
19: end
```

5.5.1.3 Business Function

Algorithm 4 specified the case where the modification is associated with updating the function. Same principle in Algorithm 1 also applies for this type of change. The difference is at the function level the impact mechanism only concerns the business process and instance active impact.

At both cases of the algorithms related to the modification of function, and the process does not involve the effects at the lower level, which is data and function respectfully. At the initial stage of the change process where the business analyst defines the fundamental requirement for function or the process, the change in the lower level has to be identified and analysed separately. For instance, if the modification is referred to the change in the function, then it is essential first to satisfy the requirement regarding the data input and output and ensure that these two types of data are available for the function to proceed. So as for this the algorithm 4 and 5 (i.e., updating function and process) we assume that the modification at the lower level is accomplished.

In this algorithm, we assume that the data for functions exist in the system. During the function modification, the impact analysis first checks the notion of compatibility. In a

case of incompatibility retrieves, all the business processes that the function f_i is included, and then check for the process integration. Finally, the same principle applies to the running instances as explained in the previous algorithm to manage the migration of process instances.

Algorithm 4: Updating Business Function:

```
1: Input: Update Busienss object f_i to f_i'
2: begin
3:     Run Compatibility Check for Business Functions ( f_i , f_i');
4:     if Business Function f_i Compatible with f_i ' then
5:       Implement New Business Function f_i ' ;
6:       Retrieve all Process using Functions f_i;
7:       for each Process Found do
8:           Retrieve all Process Instances;
9:           Migrate all Process Instances to New Business Function f_i';
10:      end for
11:      Delete Business Function f_i;
12:    else
13:      Implement New Business Function f_i ';
14:      Retrieve all Process using Functions f_i;
15:      for each Process Found do;
16:          Check Integration with other Processes;
17:          for each Integrated Processes Found;
18:              Run INTG_PRO;
19:          end for
20:          Retrieve all Process Instances;
21:          Create set for Process Instances without Process integration;
22:          Create set for Process Instances with Process integration;
23:          for each set do;
24:              Compute the critical point;
25:              if Process Instances Passed the Critical Point then;
26:                Migrate all Process Instances;
27:              else
28:                Continue the execution for all instances till passes the Critical Point
29:                Migrate all Process Instances;
30:              end if
31:          end for
32:      end for
33:      Delete Business Function f_i;
34:    end if
35: end
```

5.5.1.4 Business Process

Algorithm 5 Updating Business Process

Input: Update Busienss Process pro_p *to* $pro_p{}'$
begin
    Run Compatibility Check for Business Processes (pro_p , $pro_p{}'$);
    if Business Process pro_p Compatible with $pro_p{}'$ **then**
        Implement New Business Process $pro_p{}'$;
        Retrieve all Process calling Process pro_p;
        Retrieve all Process Instances;
            Migrate all Process Instances to New Process $pro_p{}'$;
        end for
        Delete Business Process pro_p;
    else
        Implement New Business Process $pro_p{}'$;
        Check Integration with other Processes;
        for each Integrated Processes Found;
            Run **INTG$_{PRO}$;**
        end for
        Retrieve all Process Instances;
        Create set for Process Instances without Process integration;
        Create set for Process Instances with Process integration;
        for each set **do;**
            Compute the critical point;
            if Process Instances Passed the Critical Point **then;**
                Migrate all Process Instances ;
            else
                Continue the execution for all instances till passes the Critical Point
                Migrate all Process Instances ;
            end if
        end for
    Delete Business Process Pro_p;
    end if
end

Algorithm 5 in Appendix A specified the case where the modification is associated with updating the business process. Same principle in Algorithm 1, 3, 4 also applies to this type of modification. The main difference is at the function level; the algorithm checks the impact on business processes and instances. While here, the algorithm concerns the active instance and the impact item of process integration if there is any.

5.5.2 Impact analysis algorithm: Delete Algorithm

5.5.2.1 Business Data

Algorithm 6 explains the case where modification is related to delete the data object from the system. The algorithm first checks if the data object is used by other functions in the system or not. If the algorithm captures function, then for each of the function found to check the business processes. If the algorithm found any business process, then it is not possible to delete the data object. This case only applicable if there is no business process included. Then after the algorithm only allows deleting the data object by authorising on deleting the function.

Algorithm 6: Deleting Business Data:

```
1: Input: Delete data object d_k from data object list
2: begin
3:        Retrieve all Functions using d_k;
4:        if No Functions found then;
5:          Delete Data Object d_k;
6:        else
7:          Retrieve all Process using Functions Found;    // functions from previous step
8:          if No Process found then
9:            Check Delete Functions
10:         else
11:           Print Process as Impact
12:         end if
13:       end if
14: end
```

5.5.2.2 Business Function

Algorithm 7 explains the modification related to deleting the business function. The first step is to check if the function is not used with other business processes. Then if that is the case, the next step is to check the outcome of the function if a data object is generated from the function then it is essential to check that the outcomes are not used as the input of another function. Otherwise deleting the function is not applicable since it is violating the system execution.

Algorithm 7: Deleting Business Function:

```
1: Input: Delete Busienss Function f_i from Function list
2: begin
3:      Retrieve all Process using Functions f_i;
4:      if NO Process found then
5:          Retrieve all Data objects that function f_i creating as output;
6:          if No Data Object found then
7:              Delete Functions f_i
8:          else
9:              Retrieve all Functions that using data object
10:             if No Functions found then
11:                 Check Delete Data objects;
12:             else
13:                 Print Functions Found as impact;
15:                 Print Data Object Found as impact;
14:             end if
16:         end if
17:     else
18:         Print Processes Found;
19:     end if
20: end
```

5.5.2.3 Business Process

Algorithm 8 explains the scenario where the modification applies by deleting the business process. First checks the process integration in which the business process does not serve the subprocess of another business process, otherwise it is not recommended since it creates interruption during the execution of another business process. Then if there is no integration through calling another business process, then the analyst can delete the business process by first managing the active process instance.

Algorithm 8: Deleting Business Function:

```
1: Input: Delete Busienss process pro_p from Process list
2: begin
3:      Retrieve all Process that call process pro_p;
4:      if NO Process found then
5:          Retrieve Process Istances;
6:          Flush
7:      else
9:          Print Processes Found;
10:     end if
11: end
```

Otherwise, if the instance number is more than the assigned threshold, the algorithm suggests continuing the execution until all instances are completed. The algorithm allows the

active instances to continue the execution before deleting the process. Once all the state of process instances terminated to complete then it allows the process to be deleted from the system. Note that by deleting the process we are not deleting any components in the ERP system as the functions and data objects remain in the system.

There is also another strategy that can be applied in this modification for the case when there are some active instances exist in the system. Then the business analyst, can make a decision through defining a threshold where the algorithm starts counting the number of instances and then if the number is less than the proposed threshold, then the impact analysis mechanism then allows the instances to terminate by applying the abort policies..

5.5.3 Impact analysis algorithm: Add item

Table 5-2 shows the pre- and post- conditions for addition evaluation type that is expressed as the sets constituting the ERP in the current state in which the evaluation has not yet been applied (i.e. the as-is situation type) with the one in which the modification has been implemented (i.e. the to-be situation).

Table 5-2 Addition Modification

Evaluation type	Pre-condition	Post-condition
Add-Data	$d_{k+1} \notin D$ Only if triggers from ADD-function	$D^{new} = D \cup \{d_{k+1}\}$
Add-Function	$f_{i+1} \notin F$ Only if data objects are exist in the system	$F^{new} = F \cup \{f_{i+1}\}$ $\wedge Interact\ (f_{i+1}, d_k)$
Add-Process	$pro_{p+1} \notin PRO$ Only if there functions are exist in the system	$PRO^{new} = PRO \cup \{pro_{p+1}\}$ $\wedge Uses(pro_{p+1}, f_i)$

The pre-condition for the introduction of new components ensure that the item to be added does not exist yet in the ERP system. In addition, the new process can only be introduced if all the required function available in the system and the same condition also is applied to the function in which by adding the function the required data as input and output

type must exist in the ERP system. The post-conditions also contain the predicates becoming true after the implementation of the component in the system (e.g. the new function should interact with any data object from the set *D* in the ERP system).

From the point of view of impact analysis, introducing new components, as modification type into the ERP system does not concern any effect on the current ERP components apart from the implementation of the components. Therefore, only specific action has to be taken to preserve the effort of implementation new components in the system during the change process.

5.6 Impact Metrics

A metric is a standard of measurement. It is used to judge the attributes of something being measured, such as quality or complexity, in an objective manner. A measurement determines the value of a metric for a particular object (Lorenz and Kidd 1994).

In the context of impact analysis, metrics have been primarily used for two purposes: (i) the prediction of propagation of the change in the system and (ii) the prediction of effort required to implement the change. In our case, these predictions are based on the simple notion that the more complex a piece of the ERP system is, the more likely to contain higher risk if modified.

In our framework, we define two types of metrics:

- Measure (predict/assess) of the change propagation: to quantify the impact of the entities modified at design/run time and, based on that, define a risk level for the proposed change;
- Measure (predict) of the effort: to estimate the effort of change implementation based on the cost of proposed strategies to implement the modification.

The impact metrics developed in this research provide numeric views of the effect of change, which allows the business analyst to evaluate the effect of alternative changes quantitatively.

Figure 5-10 depicts the process of defining impact metrics. It is based on three levels. Given a proposed change, the initial level (Level 0) computes the number of ERP components (business data objects, functions, and processes) and the number of running

instances (data, function and process instances) affected by the change. Then (in Level 1) the metrics of level 0 are converted into relative values, i.e. scaled against the total number of item in each category. Finally, (at Level 2) the relative impact metrics of level 1 are adjusted according to the subjective importance of each category of impact, which is captured by a weight factor. In other words, business analysts may consider that, in a given context, impact on business functions or the run-time is more/less important that impact in other areas, e.g., business processes, by assigning a higher weight factor to these categories in level 2.

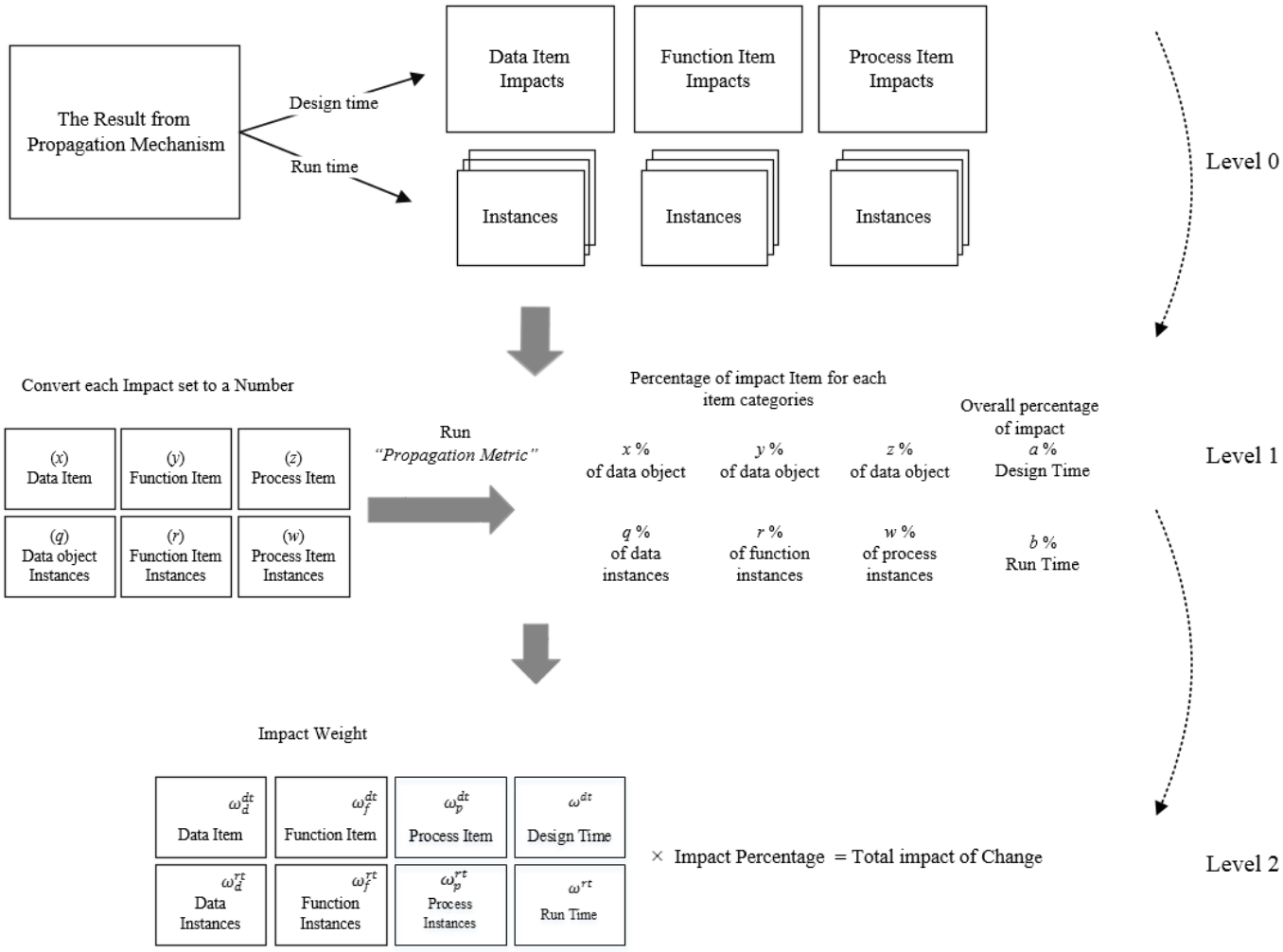


Figure 5-10 Metrics Level

5.6.1 Level 1: Impact Propagation (Relative Fraction of Impacted Items)

This section describes the metrics to measure the propagation of modification. Below the definitions of the terminology that is used in this section are given:

- DT Denotes Design Time
- RT Denotes Run Time
- *n* Denote the process (n=p), function (n=f) and data object (n=d) as ERP components
- $DT_{n=\{p,f,d\}}$ Denotes the relative fraction of impacted items at each level of change at the design time (process, function and data object)

- $RT_{n=\{p,f,d\}}$ Denotes the relative fraction of impacted items for each level of change at the run-time (process instance, function instance, data object instance)
- TPI Denotes the total relative fraction of impacted items (including both the design and runtime).

The relative fraction of impacted items is calculated according to the number of impacted entities for each level of change such as data object, business function and business process in the system divided by the total number of entities in the same categories. For example, the percentage of impacted processes is calculated based on the number of all affected business process divided by the total number of business processes in the ERP system that is:

A) Calculate the impact propagation at Design Time

$$DT_p = \frac{Total\ \ Impact\ at\ Process\ Level}{Total\ Number\ of\ Process}\ \%$$

$$DT_f = \frac{Total\ \ Impact\ at\ Function\ Level}{Total\ Number\ of\ Function}\ \%$$

$$DT_d = \frac{Total\ \ Impact\ at\ Data\ Level}{Total\ Number\ of\ Data}\ \%$$

Possibly any of impacted items at design time could have some on-going items at the run time. Therefore, it is essential to calculate the percentage of impact at the run time for each of the items from the pervious section. Same principle and technique apply to calculating the percentage of affected instances to compute the degree of change during the execution time of the ERP system.

B) Calculate the impact propagation at Run Time

$$RT_p = \frac{Total\ \ Impact\ at\ Process\ Instance\ Level}{Total\ Number\ of\ Process}\ \%$$

$$RT_f = \frac{Total\ \ Impact\ at\ Function\ Instance\ Level}{Total\ Number\ of\ Function}\%$$

$$RT_d = \frac{Total\ \ Impact\ at\ Data\ Inatace\ Level}{Total\ Number\ of\ Data}\%$$

Finally, we can calculate the total percentage of impact including both design time and run time. This informs us to what degree the impact propagates the entire ERP system.

C) *Calculate the Total impact percentage*

$$\text{Total Impact propagation } (TIP) = \frac{Total\ \ Impact\ of\ Item}{Total\ Number\ of\ Item\ at\ System}$$

5.6.2 Level 2: Relative Weight Assigned to Different Impact Categories

For ERP systems, changing a data object is likely to have a greater impact effect than changes in the function of the business process level. In fact, both functions and processes use the data object. Therefore, the ripple effects of their modification are liable to be higher. This simple consideration demonstrates that the relative importance of the impact of different types of changes is not equal. So, in our framework, we introduce weights to adjust the relative importance of the impact of different categories. In the case discussed before, the weight associated with the data object level will be much greater than the weight assigned to change the impact at the business function or process level.

5.6.3 Impact Weight Factor

Impact Weight Factor is a numeric value used to express the relative importance of impacts at different levels of an ERP system (i.e., data, function, processes) and at different stages where the impact applies (i.e., design time and runtime).

Weight factors (see Table 5-2) ω_n^s lie in the range $0 < \omega_n^s < 1$, where n is the ERP components level (data, function, object) and s is the stage where the impact propagates (i.e. design time or runtime). We also assume that the relative importance of modifications within a specific level of ERP systems (i.e., process, function and data object) are the same,

meaning that relative importance of the impact of all data object or function or processes is considered the same.

The sum of all weight ω_n^s defined in our framework equals to 1, so that a global impact metric comprised between 0 and 1 can be defined.

The business analyst normally assigns values to these weights and weights can be adjusted to fit individual requirements of specific business contexts. Values can be assigned based on common sense or personal experience, or they can be derived from surveys of domain experts and previous post-implementation modification projects. In Chapter 6, we describe a preliminary effort to collect empirical data from different ERP experts to calculate objectively the values of these weights using the Analytical Hierarchical Process (AHP) method.

Based on the above rationale we introduce a measure call Impact of Change as IMC_n^S that defines the impact of each level of change at each stage of runtime and design time and the overall impact on the system. We compute this ratio by applying AHP method and score and compare each type of impact set with another.

In definition 1 shows that the weight ratio for each level of change first and illustration on how to calculate the change impact by using the percentages which are explained in the previous section.

A. Calculate the Impact of change at Design Time

	Process	Function	Data	Total
Weight	ω_p^{dt}	ω_f^{dt}	ω_d^{dt}	$\omega_p^{dt} + \omega_f^{dt} + \omega_d^{dt} = 1$

$$Impact\ of\ change \begin{cases} \text{Process} \rightarrow & IMC_p^{dt} = \omega_p^{dt} \times DT_p \\ \text{Function} \rightarrow & IMC_f^{dt} = \omega_f^{dt} \times DT_f \\ \text{Data objects} \rightarrow & IMC_d^{dt} = \omega_d^{dt} \times DT_d \end{cases}$$

To define the Total Impact of Change at the design all the impact change for each impact set added together as follow:

$$IMC^{dt} = \sum_{n \in \{P,f,d\}} \omega_n^{dt} \times DT_n$$

The same terminology also applies to the items that impact algorithm captures as instances in the ERP system. First, compares and define the weight based on the effort of to migrate the instances. Applying modification to ERP system when the system has some running instances of data type as an impact then it is more crucial to apply migration policies compare to the function and processes.

	Process	Function	Data	Total
Weight	ω_p^{rt}	ω_f^{rt}	ω_d^{rt}	$\omega_p^{rt} + \omega_f^{rt} + \omega_d^{rt} = 1$

$$Impact\ of\ change \begin{cases} \text{Process} \rightarrow & IMC_p^{rt} = \omega_p^{rt} \times RT_p \\ \text{Function} \rightarrow & IMC_f^{rt} = \omega_f^{rt} \times RT_f \\ \text{Data objects} \rightarrow & IMC_d^{rt} = \omega_d^{rt} \times RT_d \end{cases}$$

Same as the design time the Total Impact of Change computes for the runtime by adding all the impacts together as follow:

$$IMC^{rt} = \sum_{n \in \{P,f,d\}} \omega_n^{rt} \times RT_n$$

Finally, in order to define the total impact of the change in the system including design time and run time we compare the effort of modification for both stages. We assume that the effort to migrate the ERP system at deign time is much easier to handle compare to the runtime since we might have a thousand instances running in the system and the

migration policies need to apply for each instance. Thus, the same principle applies to computing the total impact of particular modification in the system as previous parts. The total impact for each stage of design time and runtime is calculated by the assigned weight and added together.

	Design Time	Run-Time	Total
Weight	ω^{dt}	ω^{rt}	$\omega^{dt} + \omega^{rt} = 1$

$$Total\ Impact\ of\ Change = (IMC^{\,dt}) \cdot \omega^{dt} + (IMC^{\,rt}) \cdot \omega^{rt}$$

The methodology defines an impact metric that is directly related to the effort predicted to implement a change in the ERP system. The next metrics compare and provide the estimate in order to define the cost implication of the following impact in the system.

5.6.4 Metrics for Estimating the Cost/Effort Implications for Modification Implementation

Impact analysis result can be used as a measure of the cost/effort of a modification in the ERP system. The association between the metric and the effort is required to modify the system and let the business analyst to understand the effort of implementing a particular change scenario. This metric allows determine the cost implication of alternative implementation strategies that enables maintenance team to monitor their actions during ERP implementation of modification. Cost implication function is constructed in this section from the quantitative analysis of ERP expert opinion about the estimated effort required by each implementation strategy identified in section 6.5. Such a quantitative analysis is supported by the AHP method as a technique for the decision-making. Chapter 6 provides a more comprehensive explanation on how we obtain the cost estimate for each implementation strategies.

The analysis for defining the cost implication for implementation of modification in an ERP system is calculated based on the result of impact analysis algorithm in the previous

section. By first identifying the number of impact item for each modification level such that two data objects impacted, four business processes, and three business functions. Since the change, committees have a broad range of understanding of how to implement the change, then they can propose the alternative implementation strategy for each of the modification level. After identifying the proposed solution for implementation of change in ERP system, the impact analysis starts to evaluate the cost estimate. Usually, there is more than one implementation solution that can solve the same problem or satisfy the same requirement (Lee 1998). Business analysis is then computing the cost implication to measure the cost and the effort needed to implement change in their ERP system

As a result, the more the change causes other modifications of ERP system items, the higher the cost is. Carrying out this analysis before a modification is implemented allows an assessment of the cost of the change and assists management to decide the best strategies for applying the change into the system. This technique allows managers and engineers to evaluate the appropriateness of proposed modification. If a change that is proposed has the possibility of impacting a significant number of the item, the request might need to be re-examined to determine possible and feasible way to implement the modification in the system.

The calculation of cost implication of modification is distinguished in two stages of design time and run time where the impact analysis detects impact items. To quantify the impact of the cost of implication for implementation of change at the design time, it is essential to go through each item category and calculate the cost effort separately.

The metrics be based on generated propagation results and attempts to capture the overall effort for implementation in the ERP system. Some definitions of the terminology that are used in the calculation of cost implication metrics in this section are:

- ω_{s_i}= Denote the weight for implementation strategies at each level of modification
- a_{s_i} = Number of suggested strategies
- b_n = Total number of impact at each level, where n signifies on as any type of the process, function or data $\{p, f, d\}$

- EC_n^s= Denote as the estimate cost of implementation where $s = \{dt, rt\}$ as the stage which is design time or runtime and $n = \{p, f, d\}$ where n is process, function or data object.

A) Cost at Process Level

	Process Configuration	Bolt-on	ERP Programming	Workflow Programming	Interface Development
Strategies	ω_{s_1}	ω_{s_2}	ω_{s_3}	ω_{s_4}	ω_{s_5}

I. $a_{s_i} = Number\ of\ process\ for\ \ suggested\ strategy$

II. $\sum a_{s_i} = b_p$

III. $Cost\ of\ Process\ Migration = EC_p^{dt} = \sum_{S_i \in \{s_1,\dots,s_5\}} \frac{a_{s_i}}{b_p} .\ \omega_{s_i}$

In this formula first capture the number of proposed implantation by business analyst for each strategy denote as a_{s_i} where s_i signifies as the implementation strategies. The $\sum a_{i,s_i} = b_p$ indicates that the sum of all proposed change should be equal to the total impact at each level (e.g. if at the process level b_p denote as the total number of item impact than the sum of all proposed implementation strategies as $\left(a_{s_1} + a_{s_2} \dots a_{s_n}\right)$ should be equal to b_p.

In the above formula, $EC_p^{dt} = \sum_{S_i \in \{s_1,\dots,s_5\}} \frac{a_{s_i}}{b_p} .\ \omega_{s_i}$, measures the cost implication of modification as EC_p^{dt} indicate as the total effort estimate of implementation at the design time for business process. In this formula first calculate the cost of implementation for each strategy based on the number of proposed implementation (i.e. a_{s_i} that s_i refer to the implementation strategies) divide by the total number of impact (i.e. b_p) at process level. Then multiple the results with assigned weight ω_{s_i}denotes as the cost weight that assigned to

each implementation strategies. As shown in the above formula EC_p^{dt} is the sum of all cost for the proposed implementation solution.

A same technique also applies for the impact at function level, and data level. The only difference is the implementation strategies, which for each level is different from another.

B. Cost at Function Level

	Function Configuration	Bolt-on	ERP Programming	Code Modification	Interface Development	User-exits
Strategies	ω_{s_1}	ω_{s_2}	ω_{s_3}	ω_{s_4}	ω_{s_5}	ω_{s_6}

I. $a_{s_i} = Number\ of\ function\ for\ \ suggested\ strategy$

II. $\sum a_{i,s_i} = b_f$

III. $Cost\ of\ Function\ Migration = EC_f^{dt} = \sum_{s_i \in \{s_1,\dots,s_5\}} \frac{a_{i,s_i}}{b_f} .\ W_{s_i}$

C. Cost at Data Level

	Data Configuration	Query Modification	Interface Development
Strategies	ω_{s_1}	ω_{s_2}	ω_{s_3}

I. $a_{s_i} = Number\ of\ Data\ \ for\ \ suggested\ strategy$

II. $\sum a_{i,s_i} = b_d$

III. $Cost\ of\ Data\ Migration: EC_f^{dt} = \sum_{s_i \in \{s_1,\dots,s_5\}} \frac{a_{i,s_i}}{b_d} .\ \omega_{s_i}$

IV. $Total\ Estimate\ Cost\ at\ Design\ Level: EC^{dt} = \sum_{k \in \{P,f,d\}} EC_k^{dt}$

Once all the cost implication for all item at the design time are calculated then add them together to define the cost impact at the design level. This signifies that the cost estimate that requires to implement the change according to the system specification.

D. Cost at Running Instance Level

The calculation of cost implication at the run time refers to the migration policies that are applied during the impact analysis mechanism in section 5.5. As noted in section 5.4 that explains 4 migration policies of Migrate, Flush, Abort, and Flush and migrate, at the critical point for process instances detected by impact analysis. We also add another policy as manually checking the instances when modification applies. This policy is in used when the impact analysis unable to define the critical point for process instances due to the complexity of integration in business processes.

	Manually Checking	Migrate	Flush	Flush & Migrate	Abort
Strategies	ω_{s_1}	ω_{s_2}	ω_{s_3}	ω_{s_4}	ω_{s_5}

I. $a_{s_i} = Number\ of\ process\ instances\ assigned\ to\ each\ Policies$

II. $\sum a_{i,s_i} = b_{I_p}$

III. $Cost\ of\ Instance\ Migration: EC^{rt} = \sum_{S_i \in \{s_1,\dots,s_5\}} \frac{a_{i,s_i}}{b_{I_p}} . \omega_{s_i}$

E. Total Estimate Cost of Impact

The total cost estimate of effort for implantation of modification is calculated by adding both the cost at the design time and run time together as we demonstrate in the above.

$$Total\ Estimate\ Cost\ at\ Design\ Level: EC = EC^{rt} + EC^{dt}$$

The example of cost calculation is depicted in Figure 5-11. For instance, in this example if number of data item impact is 5 and if we assume that 3 of the data object can be modified through implementation and 2 through the interface development. The cost estimate is calculated the number of suggested implementation by the assign weight for each strategy at each level of modification. After adds on all calculated cost for each level to obtain the total cost of implementation.

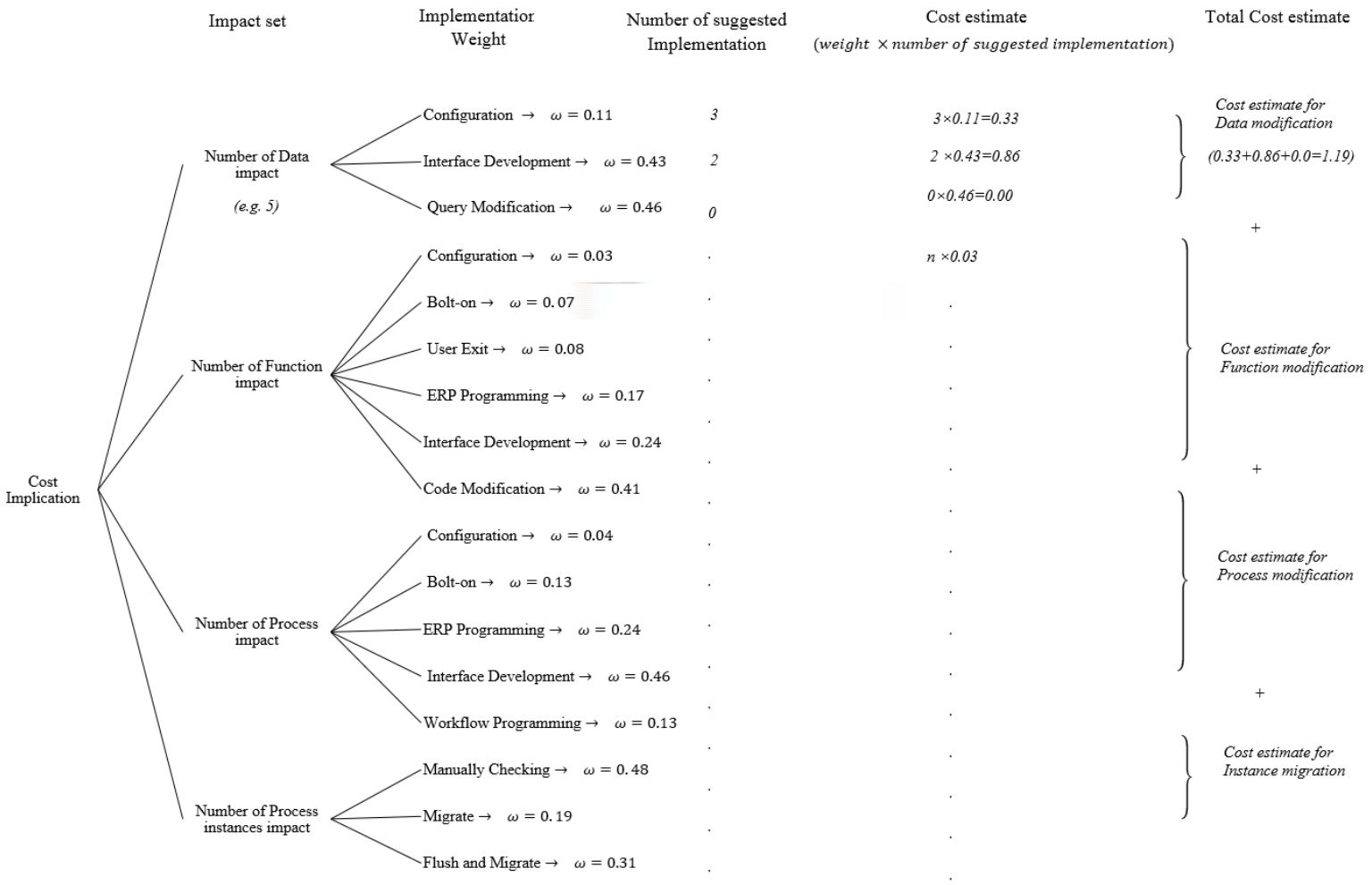


Figure 5-11 Cost Calculation

5.7 Summary

This chapter dealt with the second part of our framework, which refers to impact analysis and impact assessment of ERP modification. The initial part describes the impact analysis requisition and how the mechanism is performed in order to capture the affected item in our context. Second defines what compatibility entails in our context compared to the diverse use and overloading of the term "compatibility" in software system literature. This chapter discussed the notion of compatibility among ERP system components that we used during the impact analysis mechanism. The compatibility notion is used to compare two ERP components based on the description of the components interface and Behaviour. Then described some essential aspect of impact analysis assessment related to how to manage

running instances upon modification of any ERP components. Furthermore, described the impact analysis mechanisms, which capture the ripple effects of ERP modifications on the existing design-time and run-time structure of the ERP system. Finally, presented of metrics for assessing the effects of a proposed ERP change. These metrics aim at enhancing the decision making to plan the implementation of the modification efficiently.

CHAPTER 6 COST IMPLICATION OF ERP MODIFICATION

6.1 Introduction

ERP customization is a broad term which refers to the modifications made to the ERP system to meet the organisation's requirements that are not supported by the ERP vendor as a standard feature (Brehm, Heinzl, and Markus 2001) (Light 2001). However, many studies (Somers and Nelson 2001); (Upadhyay, Jahanyan, and Dan 2011); (Alawattage et al. 2007) indicated that the modification of ERP system is problematic and may increase costs and limit maintainability thus it is essential to plan ahead and select the most efficient implementation solution to eliminate the additional cost. As part of the impact assessment phase, our methodology also includes the planning of impact of modification implementation. At design-time, this concern estimating the cost and effort of different options for implementing the proposed change, e.g., in-house code modification versus bolt-on to a service of an external provider (Rothenberger and Srite 2009). At run-time, planning concerns identifying policies for safely completing the execution of all run-time running instances affected by the change. The cost metric during the impact assessment allows determining the cost implication of alternative implementation strategies, which enable maintenance team to monitor their actions during ERP implementation of modification. (Brehm, Heinzl, and Markus 2001) Present a typology of various ERP implementation solution explained in Chapter 2 and used in Chapter 5, but despite the benefits of having this classification, there is no evidence that defines the cost difference between these solutions. As a result, this chapter describes a preliminary effort to quantify the relative importance and effort of different types of modification strategies at the design-time to have a concrete impact assessment.

As discussed in the previous chapter while presenting the impact metrics of our framework, the implementation changes at different levels and stages can be different. For instance, implementing change through ad-hoc modification of the source code of an ERP

system implies much more effort than implementing the same functionality through simple bolt-on to an external functionality provided by a third party. While there is research focusing on the classification and impact of different ERP modification strategies, research addressing the cost implication of ERP modifications is still lacking. In this chapter, we focus on the issue of evaluating the relative cost of different strategies for ERP modification. We gather the opinion of ERP experts in ERP implementations about the relative cost of alternative strategies using an AHP-based online questionnaire. Based on the results of the data collection, we build cost functions to compare the relative cost of alternative strategies. Our approach represents a first step in quantifying the relative cost of different ERP implementation strategies, and we deployed the result of measuring the impact metrics in Chapter 5. Furthermore, the result can be exploited in decision-making problems related to ERP change and evolution of selecting the appropriate strategies for implementation of the new requirement in the ERP system.

This chapter is structured as follows. Section 6.2 introduces the research background and related work about ERP modifications and provides a classification of different strategies for ERP modification. Furthermore, Section 6.3 presents an overview of the AHP decision-making technique. In Section 6.4, we design our AHP-based technique to construct a cost estimate of ERP modification. In Section 6.5, we discuss and analyse the result of our study whereas Section 6.6 summarises our contribution and outlines the future work.

6.2 Classification of Different Strategies for ERP Modification

In most cases, organisations are going through the modification of ERP systems without fully understanding and comparing the cost implications of the selected solution (Rothenberger and Srite 2009). Estimating the cost of different strategies for implementing ERP modifications is an open research problem. In this paper, we present a preliminary empirical investigation of the cost of different ERP modification implementation strategies based on the Analytical Hierarchical Process (AHP) (Saaty and Vargas 2012). In particular, we first classify the type of changes that can occur in ERP systems and then develop an AHP-based technique that facilitates decision-making to identify the relative cost of different ERP modification strategies. Our approach is based on the design of a questionnaire answered by

ERP experts to compare and rate the modification options according to the cost of implementation for each level of change.

As far as ERP modifications are concerned, (Luo and Strong 2004) report that, if approaches and frameworks for evaluation of customization choices are employed, then these would help organisations to take decisions about customization during ERP implementation. In the past decade, research effort has gone towards understanding the reasons for customization (Rothenberger and Srite 2009) (Zach and Erik Munkvold 2012) (van Beijsterveld 2006). (Luo and Strong 2004) Were the first to propose a framework that unites the business processes of the enterprise with their ERP system. The purpose of this framework is to allow organisations to understand and identify the most efficient customization solutions, according to the availability and feasibility of adaptation. More specifically, (Brehm, Heinzl, and Markus 2001) develops a topology of technical customization categories that reflects essential aspects of ERP tailoring (on the application, communication and database layer).

Work (Akkiraju and van Geel 2010) analyses ERP customization from a cost estimation perspective by combining an artefact-centric approach and linguistic analysis approach. Later the author (Ng 2012) develops a model for predicting the benefits and cost of subsequent maintenance and upgrades to the system. Particularly about cost implications, research such as (Huang et al. 2004) (Ng 2012) (Akkiraju and van Geel 2010) (Parthasarathy and Daneva 2014) evaluates the risks associated with customization decisions during ERP implementation. All the studies mentioned above do not provide an empirical validation of the proposed model and, most importantly, do not consider different implementation strategies for ERP modifications.

There are different types of change that have been studied in the enterprise systems and business process management literature (Kherbouche et al. 2013) (Mento, Jones, and Dirndorfer 2002). According to (Soh, Kien, and Tay-Yap 2000) (Shiang-Yen, Idrus, and Wong 2013), there are three types of modification that may occur within ERP systems to handle misfit issues, such as the change in data or output, business processes, and functional change. Modification of business data arises when there is an incompatibility in the data format and relationship in the data model (Soh, Kien, and Tay-Yap 2000). Functional change may result from incapability in the execution of the business data (Yen, Idrus, and Yusof

2011). Finally, modification of business processes is devoted to the improvement in the execution of business activities in ERP systems to model the workflow of the organisation.

For each proposed change, there are different implementation strategies available. In the literature, the work of Bream address strategies on how to implement ERP modifications (Brehm, Heinzl, and Markus 2001). Based on the description addressed in Chapter 2 about the ERP customization strategy, not all strategies applied to the types of changes. Table 6-1 explains the relevant appropriateness of the identified strategies for each type of change (Parhizkar and Comuzzi 2015).

Table 6-1 Implementation Strategies for each type of Change

Type of Change				
	Change type	***Data***	***Function***	***Process***
Configuration	Add-Update-Remove	✓	✓	✓
Bolt-on (add-on)	Add		✓	✓
User-Exits	Update		✓	
ERP Programming	Add		✓	✓
Code Modification	Add-Update-Remove		✓	
Workflow Programming	Add-Update-Remove			✓
Query Modification	Add-Update-Remove	✓		
Interface Development	Add	✓	✓	✓

The next section provides an overview of a technique namely as AHP in order to compare these strategies according to the cost/effort of implementation during the modification of ERP systems. A variety of research has employed the AHP method for assisting decision-making in the area of ERP selection, ERP maintenance risk assessment, and requirements-based ERP customization (Parthasarathy and Daneva 2014) (Sarfaraz, Jenab, and D'Souza 2012) (Huang et al. 2004) however no evidence found that shows the comparison between these strategies based on the cost of implementation.

6.3 AHP Methodology for Decision-Making

Analytic Hierarchy Process (AHP) (Saaty 2003) is multiple criteria decision-making (MCDM) methods, widely applied by practitioners and researchers to prioritise alternatives based on a set of evaluation criteria. The method allows estimating the relative importance of

the evaluation criteria first, and then to prioritise the alternatives. AHP calculates priorities based on the elicitation of pairwise comparisons from experts, on both criteria and alternatives, under the assumption that experts are more reliable when asked to compare only two alternatives or criteria, as compared to three or more.

The AHP method comprises the following steps: first a complex decision problem is decomposed in a hierarchical model of goals (criteria) and alternatives (See Figure 6-1); second, pairwise comparisons of alternatives at each level of the hierarchy are performed; finally, after having checked the consistency of the results, the judgments are synthesized to support decision making (Saaty 2003).

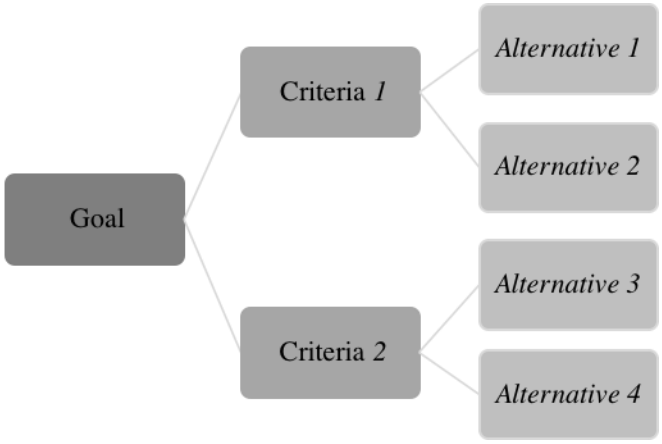


Figure 6-1 Representation of Hierarchical Model

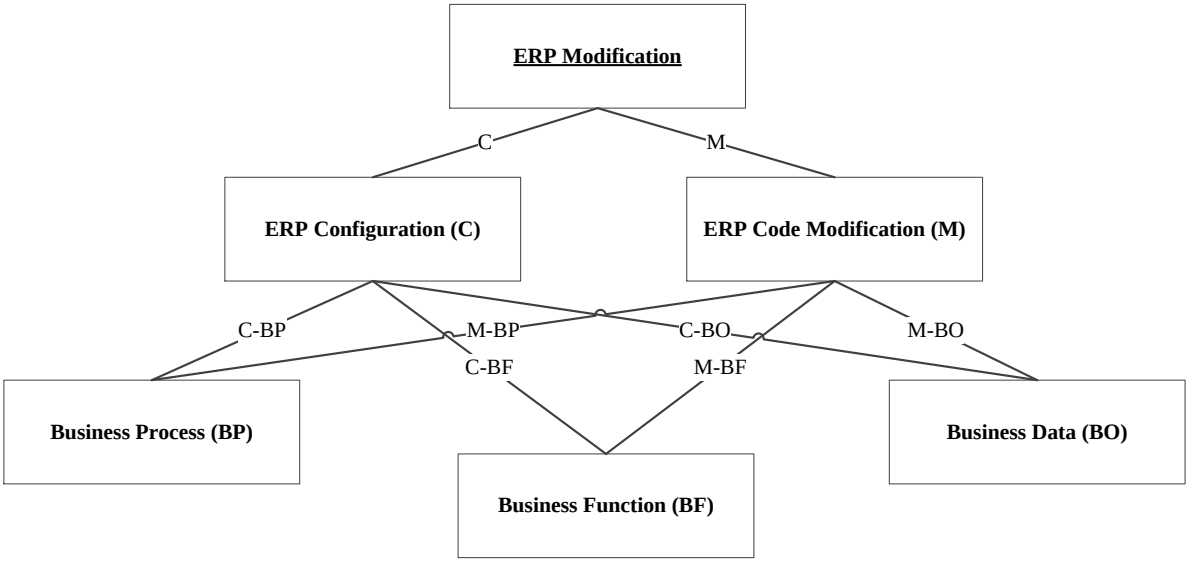
6.4 Design Empirical Study

When faced with the implementation of a change in an ERP system, business analysts are required to understand and estimate the cost of the proposed modification to plan the implementation more efficiently. As we discussed before, researchers have tackled the problem of the cost of ERP customization, but none is focusing on the cost comparison among alternative modification strategies. The AHP method could resolve this problem, by allowing business analysts to prioritise the ERP customization strategies (indicated in Table 6-1) based on their cost implications.

Three steps constitute our study:

- Development of the hierarchies and alternatives, i.e. ERP modification implementation strategies;

- Empirical survey of ERP experts to assess the relative importance of identified alternatives, based on AHP;
- Creation of empirical cost functions of alternatives of ERP cost modifications to support decision-making based on the priorities resulting from the application of AHP.



- C: ERP Configuration
- M: ERP Code Modification
- BP: Business Process
- BF: Business Function
- BD: Business Data

Figure 6-2 Hierarchy Model based on configuration and code modification

The first step toward the design of this research study is to structure the problem in the form of a hierarchy. We developed two hierarchical models as illustrated in Figure 6-2 and Figure 6-3. The first one evaluates the comparison of configuration and customization in ERP systems, according to the different level of change (i.e. Business Process, Function, and data). The second hierarchical model considers the different types of strategy that can apply to each level of ERP modification.

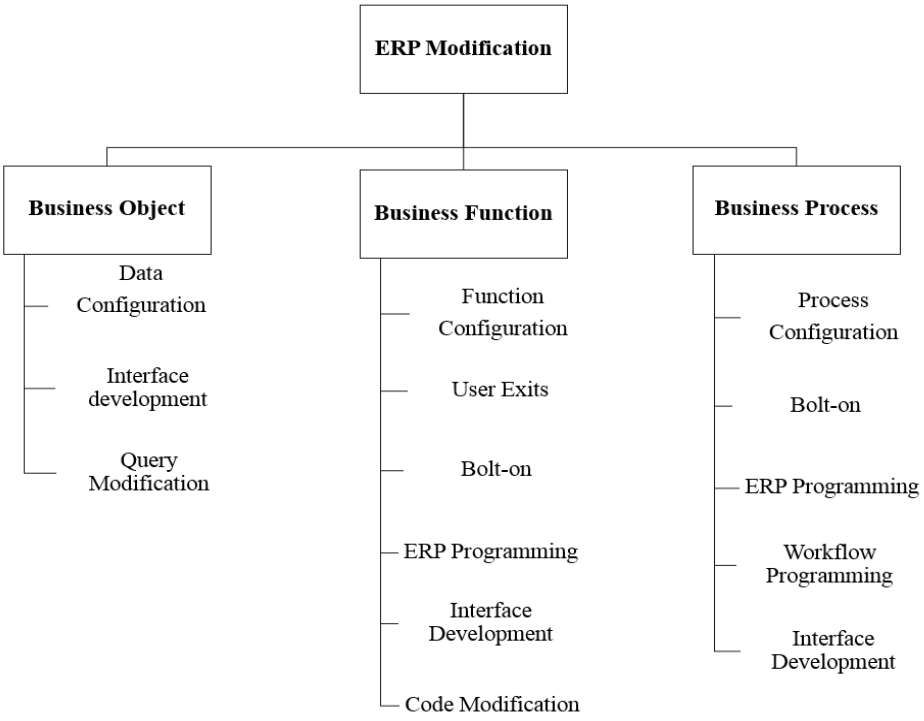


Figure 6-3 Hierarchy Model of overall strategies

In the second step, we developed surveys in the form of an online questionnaire to allow ERP experts to compare the criteria and alternatives identified by the hierarchies of Figure 6-2 and Figure 6-3 (see Appendix B). A total number of 100 experts, i.e. IT Managers, senior ERP consultant, and ERP developers were invited from the ERP Change Management group in LinkedIn to fill in the survey (See Figure 6-4). A total of 28 responses were eventually considered for analysis.

In our comparison, following the AHP best practices, respondents can capture the relative importance of alternatives in a hierarchy assigning numbers between 1 and 9 in pairwise comparisons. The assigned value 1 corresponds to the case where two modification strategies are considered equally expensive to implement a certain modification. The number 3,5,7,9 correspond to the case in which one of the two modification strategies is considered more expensive in terms of implementation, and the even number represents intermediate values (See Table 6-2). An example of the possible comparison is provided in Figure 6-5 as a sample as we used in our questionnaire (See Appendix B)

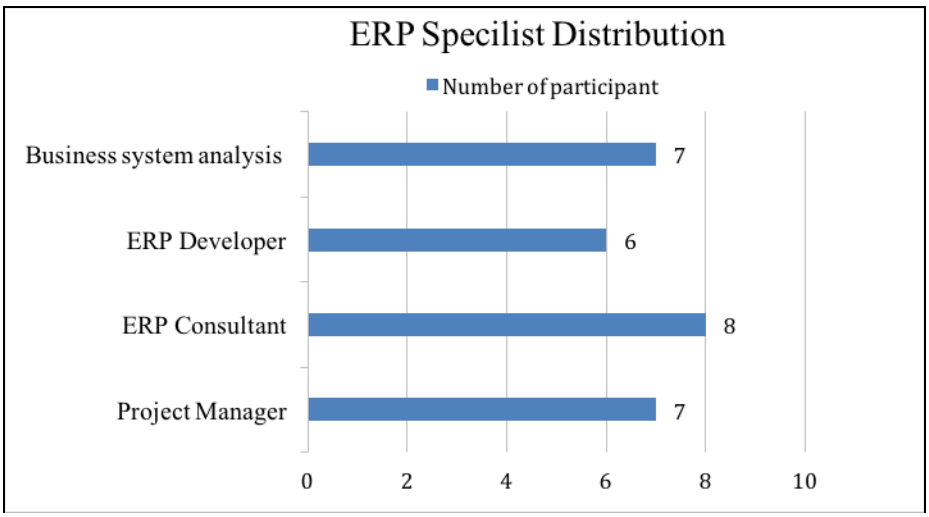


Figure 6-4 Distribution of ERP Expert

Table 6-2 Pairwise comparison Scale

Verbal judgment	Numerical Rating
Equally Expensive	1
Slightly Expensive/Cheaper	3
Moderately Expensive/Cheaper	5
Extremely Expensive/ Cheaper	7
Very Extremely Expensive / Cheaper	9
Intermediate Value	2,4,6,8

The questionnaire was structured in three sections: the first section ascertained the participant experience in ERP change management. The second part concerned the prioritisation of the alternatives in the hierarchy of Figure 6-2. The last part asked participants to rate the relative cost of alternatives of the same type of strategies, i.e. configuration and code modification, among three levels of change in ERP systems in the hierarchy of Figure 6-3. This is to scale the magnitude of cost for different levels of change while building our cost functions (see next section for more details).

The analysis of the results collected through the survey and the development of cost functions is discussed in the next section.

Q1) Please Compare Rows with Column for each Type of Modifications at Functional Level.

For example:
*"Bolt-on type of modification for ERP functions is **Moderately** more expensive than Configuration"*

	.... than Configuration		Scale From 1-9
	Expensive	Cheaper	
The Bolt-on is	●	○	5
The User-Exits is	●	○	4
The ERP-Programming is	●	○	6
The Interface Development is	●	○	7
The Code modification is	●	○	9

	.... than Bolt-on		Scale From 1-9
	Expensive	Cheaper	
User-Exit is	○	●	1
ERP-Programming is	●	○	4
Interface Development is	●	○	6
Code modification is	●	○	8

	... than User-Exits		Scale From 1-9
	Expensive	Cheaper	
ERP Programming is	●	○	
Interface development is	●	○	
Code Modification is	●	○	

Figure 6-5 Example of Questionnaire

6.5 Analysis of the Results

In this section, we present the data collected through our survey, and we discuss how, starting with those results. We created cost functions to compare different ERP modification implementation strategies.

6.5.1 Normalization

According to AHP practice, participants' responses were transformed into comparison matrices (one matrix for each evaluated hierarchy). Then, for each matrix, the priorities were calculated as the principal eigenvector (Saaty 2003). Figure 6-6 shows an example of the matrix result from one of the participant judgment that compares the implementation strategies for ERP function modification. The figure 6-6 shows the weight using the principle of eigenvector on the right side.

Matrix		Configuration	Bolt-on	User-exits	ERP Programming	Interface Development	Code Modification	normalized principal Eigenvector
		1	2	3	4	5	6	
Configuration	1	1	1/4	1/4	1/6	1/7	1/9	2.60%
Bolt-on	2	4	1	1/2	1/4	1/4	1/7	5.53%
User-exits	3	4	2	1	1/3	1/5	1/5	7.41%
ERP Programming	4	6	4	3	1	1/2	1/5	14.78%
Interface Development	5	7	4	5	2	1	1/4	21.84%
Code Modification	6	9	7	5	5	4	1	47.85%

Figure 6-6 Sample of Matrix result for one

6.5.2 Consistency Check

The important constraint in this process is the consistency of pairwise judgment provided by the decision makers (Saaty 2003). The consistency ratio is designed in such a way that a value greater than 0.10 indicates an inconsistency in pairwise judgment. Thus, if the consistency ratio is equal to 0.10 or less, then the consistency of the pairwise comparison is reasonable. Otherwise, the provided comparison can be adjusted to restore the consistency. Then in Figure 6-7 present the example of inconsistency in the pairwise comparison, which is required to be excluded from analysis. Such an adjustment, however, should not completely reverse the opinion of the decision maker.

Table

	Criterion	Comment	Weights	Rk
1	Configuration		2.6%	6
2	Bolt-on		5.5%	5
3	User-exits		7.4%	4
4	ERP Programming		14.8%	3
5	Interface Developme		21.8%	2
6	Code Modification		47.8%	1
7			0.0%	
8			0.0%	
9		for 9&10 unprotect the input sheets and expand the	0.0%	
10		question section ("+" in row 66)	0.0%	

Result

Eigenvalue			lambda:	6.442	
Consistency Ratio	0.37	GCI: 0.26	CR:	7.0%	

Figure 6-7 Consistency Check

6.5.3 Aggregate the Responses

There are several possible ways, introduced by (Escobar and Moreno-jiménez 2007), to aggregate the responses given by different experts in AHP: (ISO-9126) aggregating the individual judgments (AIJ) for each set of pairwise comparisons into an aggregate hierarchy; (ISO-9126) synthesizing each of the individual's hierarchies and aggregating the resulting priorities (AIP – aggregating individual priorities); and (3) aggregating the individual's derived priorities in each node in the hierarchy. In our work, we considered AIP, which is more suitable for aggregate comparisons expressed by participants with heterogeneous skills. After calculating the consistency index and adjusting for minor inconsistencies, a total number of 20 responses were retained for analysis. A total of 8 responses, therefore, were not considered further because they were showing great inconsistencies. Therefore, the results obtained using AHP and AIP method is presented in Figure 5-8.

a) Normalized Principle of Eigenvector of First Hiraracical Model (Figure 1)

1) Configuration:

Business Object	Business Function	Business Process
38.8	37.2	24.1

2) Code Modification:

Business Object	Business Function	Business Process
31.3	43.5	25.3

b) Normalized Principle of Eigenvector of Second Hiraracical Model (Figure 2)

3) Business Object Modification Level:

Data Configuration	Interface Development	Query Modification
11.1	43.2	45.7

4) Business Function Modification Level:

Function Configuration	Bolt-on Functionality	User-exits	ERP Programming	Interface Development	Source Code Modification
3.3	6.7	7.7	17.3	24.3	40.9

5) Business Process Modification Level:

Function Configuration	Bolt-on Functionality	Workflow Programming	ERP Programming	Interface Development
4.4	12.9	13.3	23.6	45.8

Figure 6-8 Weight Priorities

To build cost functions from the calculated priorities, we first needed to compare the relative perceived cost of the same implementation alternative at different levels of ERP modification. In other words, the last part of our questionnaire asked the participants to rate the cost of the same alternative, i.e. configuration or code modification, at the business object, business functions, and business process modification level. Note that such a comparison could work only on configuration and code modification since they are the only implementation strategies that apply to all levels. The result of such a comparison is given in figure 6-8 (a). Based on the result of the priorities presented in Figure 6-8 (b) are rescaled to obtain indicative cost functions for implementation strategies at the different level of the ERP system. Results after rescaling are shown in figure 6-9.

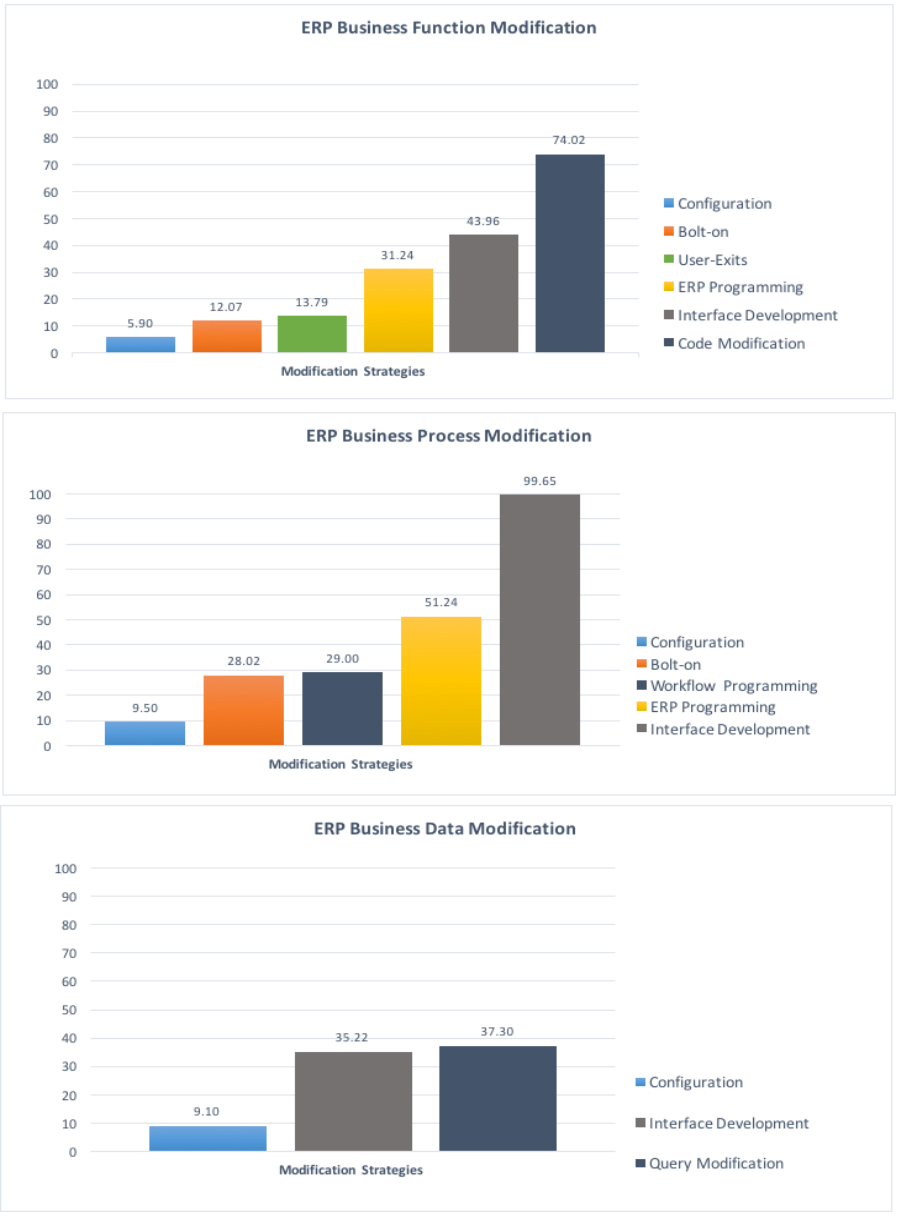


Figure 6-9 Final Cost Results

6.6 Discussion

The information is captured by our cost functions can help the business analysts to measure and identify the difference between different alternative strategies for ERP modification. For instance, in the case the same function can be accessed externally (through bolt on), implemented by extending the system (i.e. ERP programming), or developed ad-hoc (code modification), our results give a clear estimation of the cost implied by each of the

three options. Our results also show that the perceived cost of different alternatives does not grow linearly, but it follows a specific profile at each level of ERP modification.

The purpose of this research was to evaluate the perceived cost of different ERP modification strategies. We apply the AHP decision-making technique to elicit the knowledge of ERP experts in assessing the relative cost of implementation alternatives and we developed a set of cost functions to assist practitioners in selecting the most appropriate implementation strategy for a given ERP modification.

Our research approach contributes to extend the related work in the field of cost analysis of ERP customization. The work by (Luo and Strong 2004) has been taken as a foundation of ERP modification implementation options. In addition, the work by (Brehm, Heinzl, and Markus 2001) introduces nine types of customization solution without discussing the cost implication of ERP systems. Our work complements such literature, by consolidating more extensive knowledge to compare, in terms of cost implications, various solutions in order to handle modification in ERP systems.

Our work is clearly in progress and can be refined by collecting more data, e.g. involving a larger group of ERP expert with dynamic skills to, e.g. developers, consultant, and manager, to increase the reliability of the results. Most importantly, our results should be validated in real ERP modification scenarios. At the same time, however, we argue that our work can be helpful in any context related to decision-making about ERP change and evolution.

First, we plan to use our cost functions in the context of an ERP modification impact analysis framework that we are concurrently developing. When different strategies for implementing a given change are available, our framework will be able to rank alternatives based on both their impact on the ERP system as a whole and their implementation costs.

Exploring more customization option and adjusting our cost functions to different types of ERP system, such as on premise and cloud ERP, will also be interesting directions for future work.

6.7 Summary

As discussed before while presenting the impact metrics of our framework, the impact of a proposed change at different levels and stages can be different. For instance, implementing change through ad-hoc modification of the source code of an ERP system implies much more effort than implementing the same functionality through simple bolt-on to an external functionality provided by a third party. In this chapter, we described a preliminary effort to quantify the relative importance and effort of different types of impacts and modify strategies. In particular, it provided more detail on the parameterization of functions to estimate the implementation effort of ERP changes. We gathered the opinion of experts in ERP implementations about the relative cost of alternative strategies using an AHP-based online questionnaire. The results presented in this chapter have been used to improve the impact assessment metrics in the software tool implementation to enhance the decision-making of applying a change in ERP system. Based on the results of the data collection, we built a cost function to compare the relative cost of alternative strategies.

CHAPTER 7 PROOF OF CONCEPT IMPLEMENTATION

7.1 Introduction

This chapter gives more insight into the design and implementation of the impact analysis tool as proof of concept. An agile development methodology and a model-driven approach were chosen as the foundation of our implementation. First, we present the design and implementation of the method for developing the impact analysis tool. Then we provide detail about the tool implementation to support our methodology and, finally, by showing a running example we demonstrate the feasibility and applicability of our tool.

This chapter first reviews on the software development methods and explains the reasons for conducting agile methodology during the implementation of impact analysis in Section 7.2. Then in Section 7.3 provides a detail description of the requirements including the features and functionality that our tool can support. Section 7.4 discusses the design and implementation specification as a model-driven approach and demonstrates the implementation using Mendix, a model-driven development tool. Section 7.5 provides a walkthrough with the example to demonstrate how the impact analysis works in our tool.

7.2 Software Development Process:

According to (Jacobson et al. 1999), software development is the set of activities that are expected to result in software products. The software development activities must be divided into several different parts so that development flow is preserved. Methodologies require a disciplined process in software development with the intention of making software development more efficient and productive. Typical software development methodologies are the Waterfall, Agile, RAD, Prototyping, and Spiral.

Typically, software development methodologies are divided into heavyweight and lightweight. The heavyweight methods, considered as the traditional way to design software, require a comprehensive planning, detailed documentation, and expensive design. Heavyweight methods are plan driven in which the development begins with the elicitation

and documentation of a comprehensive set of requirements, followed by architecture and high-level design and inspection. The steps in these methods are based on a consecutive series, such as requirements definition and analysis, design and solution development, testing and deployment (Leau et al. 2012). Heavyweight methodologies require defining and documenting a stable set of requirements at the beginning of a software development process.

The three most important methods of this type are Waterfall, Spiral Model, and the Unified Process. The waterfall approach indicates a structured progression between defined phases. All efforts, including modelling, are formed into workflows in the Unified Process (Kruchten 2004) and are performed in an iterative and incremental manner. The spiral model combines elements of both design and prototyping, to assist advantages of top-down and bottom-up approaches.

The lightweight methods, known as agile methodology, have gained significant attention from the software engineering community in the last few years. Compared to the traditional methodologies, the agile method applies to short iterative cycles and relies on an implicit understanding within a team.

An Agile methodology is a method to software project management that contributes to respond to the unpredictability of developing an application through incremental and iterative work. Agile development methodology provides the opportunity to assess the direction of our impact analysis throughout the development lifecycle based on iterative cycle to build and test until it satisfies our objectives. (Qumer and Henderson-Sellers 2008) Pg. 1900, described agile development as

"A software development method is said to be an agile software development method when a method flexible, (ready to adapt to expected or unexpected change at any time), speedy (encourages rapid and iterative development of the product in small releases), lean (focuses on shortening timeframe and cost and on improved quality)."

7.3 Requirement Specification

7.3.1 Impact Analysis Functionality and Development Iteration

An agile based method meaning, iteration-based development and a feature driven. Iteration can be thought as the stage during software development where the developer can check and test the system features and get the feedbacks about the Behaviour. The feedback can provide information emphasis on the operations on whether the system works as it supposes to be or not.

The previous chapters have presented an understanding of the domain problem and the design of our framework. The framework has been embedded in a software tool (i.e., a decision supports system) to demonstrate the feasibility of our approach for the business analyst during ERP post-implementation change. The design of impact analysis tool has been divided into three phases: the ERP dependencies model that explains the interaction between existing ERP system components; the impact analysis that involves the mechanisms of different change operation; and the impact assessment to quantify the impact of proposed change based on the result of impact analysis. Successful development of our impact analysis tool entails that the requirements are stated clearly for each phase, to understand how the impact analysis tool works. Therefore, it is important to identify the main features and functionality of the tool by defining the main requirements. According to the above phases, the following list classifies the fundamental requirements that we aimed to be achieved during the development of our Impact Analysis tool.

- The initial requirement is to create the dependency repository this should be defined by the business analyst to extract the information from the ERP system and mapped them accordingly to the impact analysis tool. Business analysts can then explore this information about the design-time components, e.g., a list of business processes, functions, modules and data objects. The tool also needs to facilitate the users to create as many process instances available in the ERP system and allows them to check and define the status of running instances. This facilitates the definition of the ERP dependency model imposed by our framework.

- For impact analysis, the process begins with performing the impact mechanisms according to the change operations verified by the user from the change request form to find the affected components through the analysis of dependency repository.
- The last requirement is that impact analysis tool access the information about impacted items and synthesises into a compact set of impact assessment metric for decision makers. During the assessment, the impact analysis tool should be able to give an estimate of the effort associated with different strategies for implementing the change. The tool also needs to provide a basic dashboard to graphically compare the impact of the different change request regarding a number of affected components and effort estimation.

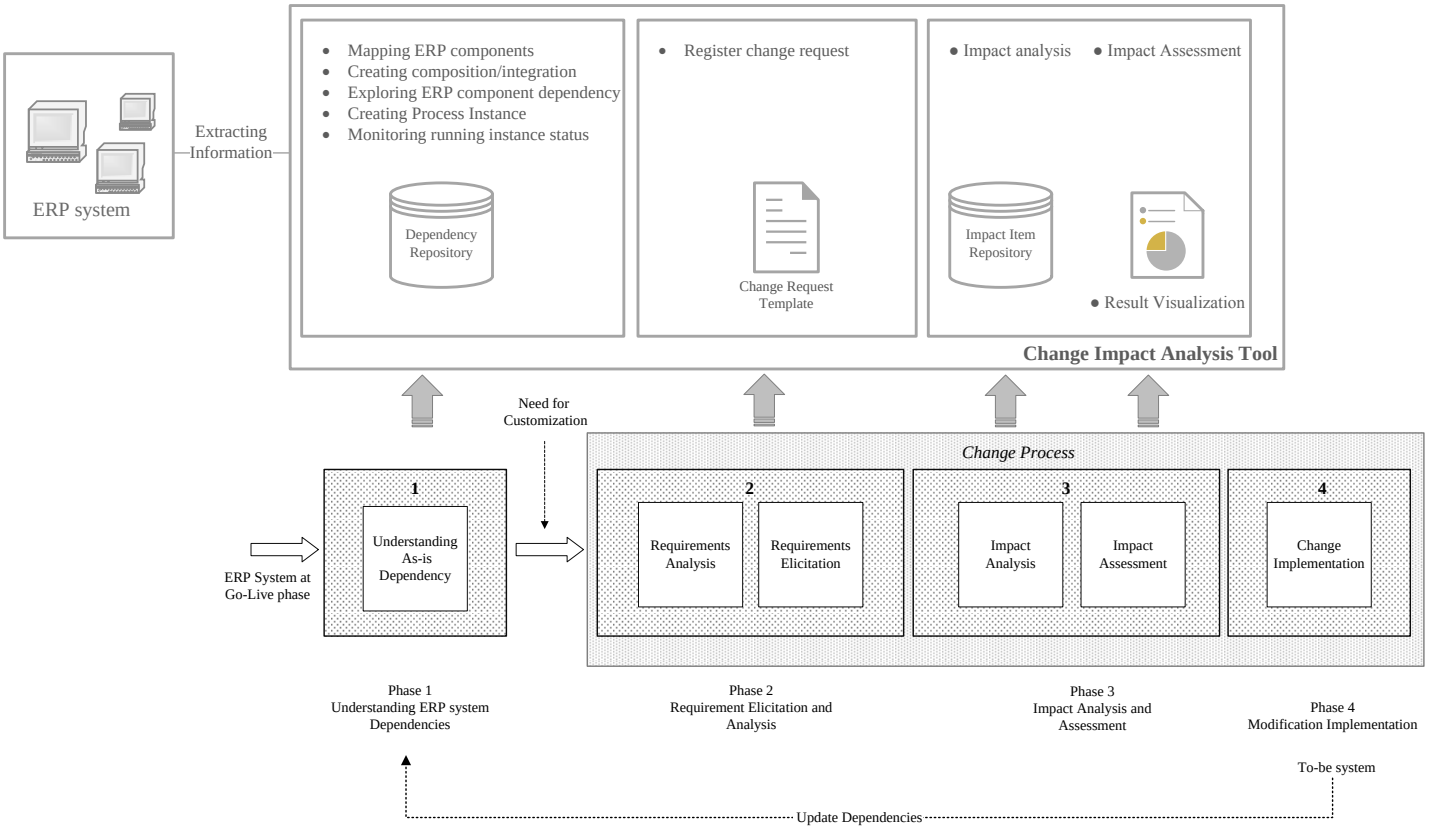


Figure 7-1 Representation of impact analysis tool based on change process framework

To demonstrate the above requirement, we outline all activities required to develop into the impact analysis tool. Besides, Figure 7-1 illustrates the association between these activities at each phase of our framework.

1) Mapping ERP components: the analysts can define and assign all ERP system components (i.e. process, function, data, module) and creates the dependency relation between them.
2) Composition/Integration: the analyst can design the process integration in the tool by composing different business process together to demonstrate the complex cases for analysis. (e.g., the production process uses procurement process as sub-processes when purchasing the raw material).
3) Exploration: Allows the analyst to observe and search the ERP components throughout the system. Furthermore, this functionality provides information about basic requirements and dependencies with other components when analysing the requirements (e.g. the tool can confirm which functions are using the data object as input and output while searching for a particular data object).
4) Creating Process Instance: the analysts can specify a process instance for a particular transaction in the ERP system. Process instance creation enables to create an instance of a different kind such as functions and data that is associated with business processes.
5) Monitoring running instance status: the business analyst can monitor and update the state of each process instance.
6) Register change request: the business analyst can create and store various types of change at different level and types according to the taxonomy introduced in the previous chapter. This is achieved through filling the change request template while analysis the ERP end user requirement.
7) Impact Analysis: when business analyst runs the impact analysis the tool can evaluate the dependency on different components concerning data, functions, processes, and instances, etc. and classifies them into a report. Also, the tool can define what actions the analyst has to take during the migration of running instances.

8) Impact Assessment: Measures the depth of impact through the risk assessment metrics defined in Chapter 5 and estimates the effort and cost required for change implementation.
9) Result Visualisation: the tool should provide a representation of impact assessment for all changes requests to compare the number of affected components and effort estimation to enhance a better decision for planning the implementation.

Figure 7-2 demonstrates the main use case of overall impact analysis tool. The proposed tool is developed in three iterations; each explained in detail in next three sub-sections.

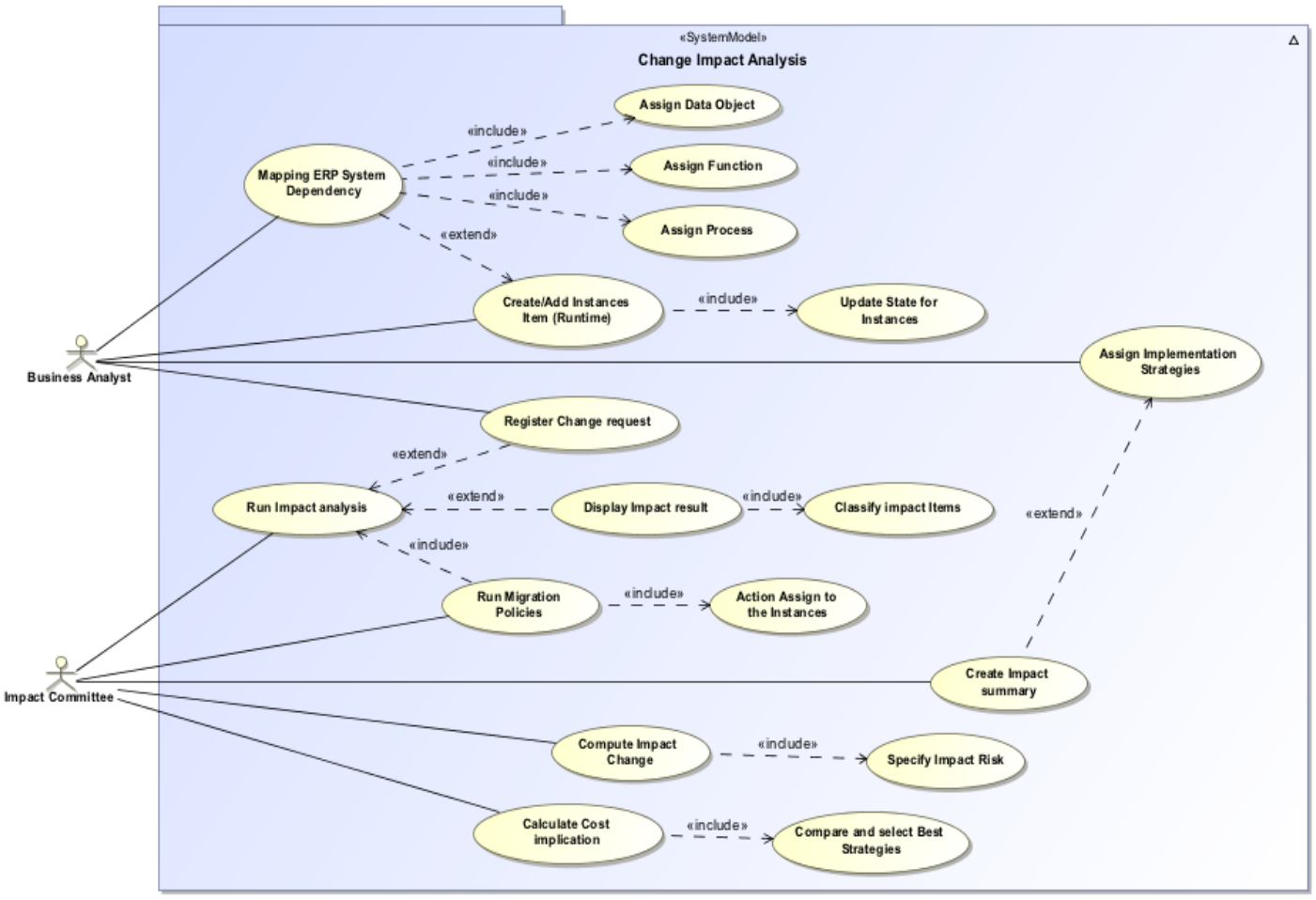


Figure 7-2 Use Case of Impact analysis tool

7.3.1.1 First Iteration

The first iteration is illustrated in Figure 7-3, and it concerns all the requirements to preserve ERP components dependencies during the execution of the impact analysis tool. This includes the development of ERP dependencies meta-model explained in Chapter 4. At this stage of development, the tool allows the business analyst to map the ERP system components to assess the impacts. The tool also facilitates the creation of instances that represented at the business transaction in ERP system during execution time. This also allows the user to monitor the status of the process instances and update the state of progress until the completion. During the creation of process instance, the impact analysis creates instances of all the components included in the process model (i.e. function and data).

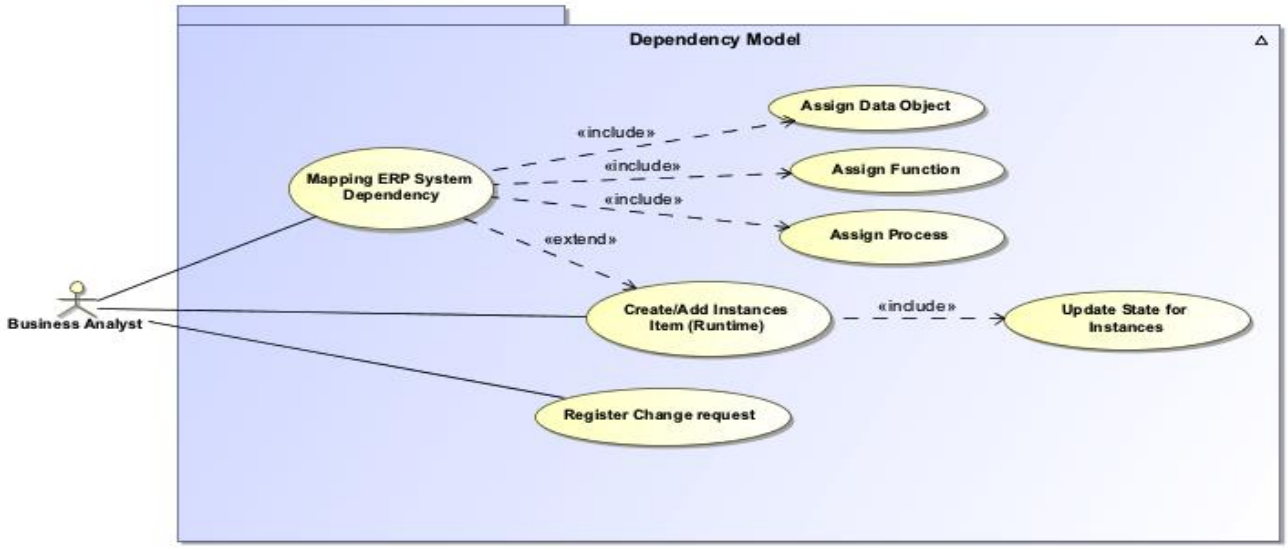


Figure 7-3 First Iteration ERP Dependency Model

7.3.1.2 Second Iteration

The second iteration shown in Figure 7-4 is concerned the implementation of the impact analysis mechanism. In order to preserve the impact of modification, it is important to define the characteristic of modification by creating the change request and evaluate it based on the level and type of modification. Then, according to the request specification, the tool implements the related mechanism explained in Chapter 5 to run impact analysis. Furthermore, in this part of the implementation, the tool needs to apply the migration policies on the active instances that are identified during the execution of impact mechanism.

Once all impacted item capture for both design time and run time the tool should generate the report summary indicating the number of items impacted in the system. The report then measures the propagation of impact according to the degree of complexity and the importance of the modification items (See Chapter 5). Then based on the result of propagation establishes the level of modification risk.

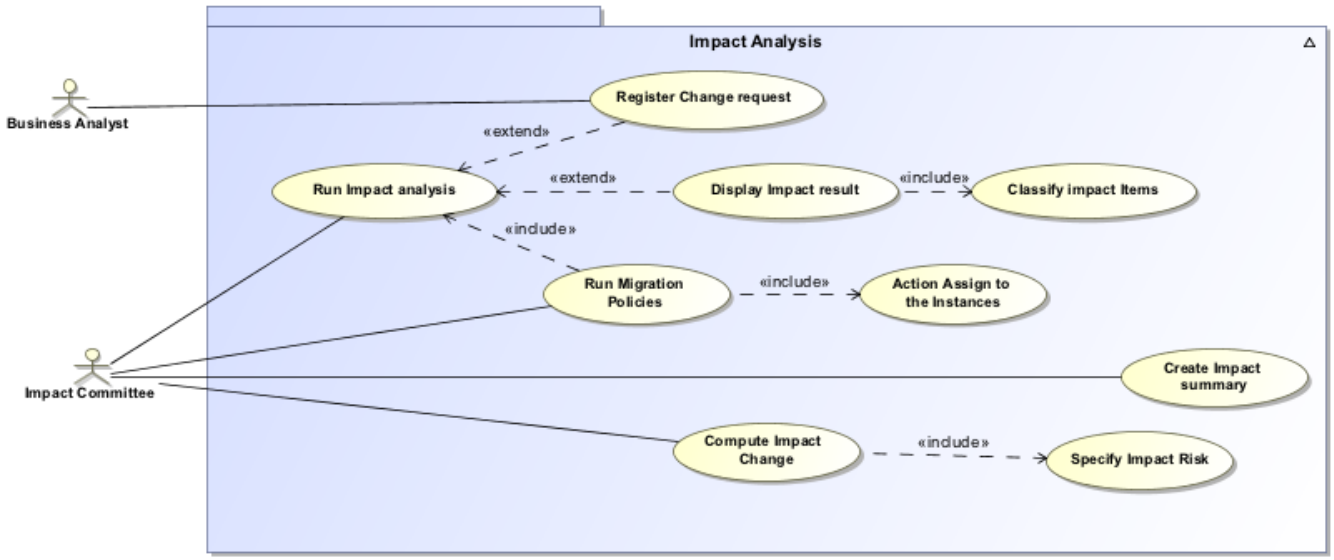


Figure 7-4 Second Iteration Impact analysis

7.3.1.3 Third Iteration

The third iteration described in Figure 7-5 focuses on the planning of implementation as a result of the modification. In this stage, the impact tool asks the business analyst to assign the implementation strategies for predicting the cost/effort implication of the change. Based on the number of impact items for each category from the report in the previous iteration, the user can propose many possible solutions for a particular request and compares them to make a decision about which strategies bring the less cost for implementation. The visualisation compares the cost of implementation among impact items categories such as data, functions, processes and running instances.

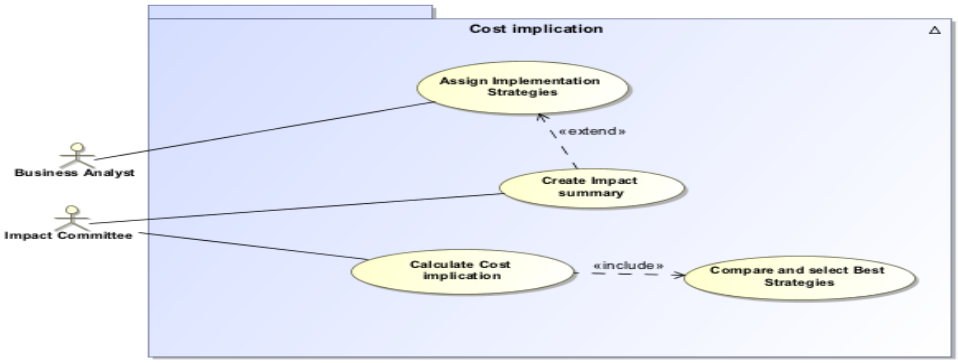


Figure 7-5 Third Iteration Cost Implication

7.4 Design and Implementation Specification

7.4.1 Model Driven Approach

One of the latest technologies that have been established in the past few years is the model-driven approach. Current trends such as Model-Driven Architecture (MDA) (Soley 2000) or other model-centric approaches are positioning modelling under the attention more than ever before (Bézivin 2006). Model-driven engineering is the development of software based on the use of models, which can develop application much faster than a classical coding implementation. The model-driven approach provides the designer with an easy way to abstract and visualise the application. This method makes use of executable code either in a form of a high-level programming language or by direct implementation of the model.

In order to design and implement our impact analysis tool, we selected the Mendix platform to accomplish our objective. The next section will discuss the reasons for choosing Mendix (www.mendix.com) a model-driven application and further explains the features of this application that makes it easier to develop our impact analysis tool for this research purpose.

7.4.2 Mendix: A Model-Driven Application Development Tool

Mendix application was established in 2005 and started as a spin-off from the Technical University of Delft and the Erasmus University of Rotterdam. This application has been recognised as a leader compared to the other applications that support a visual and

model-driven approach of developing a software tool. Mendix provides a model-driven enterprise application platform that enables business analysts to develop service-oriented business applications that can be integrated and adapted to many existing IT & business environments. Main advantages are increased flexibility, accelerated application delivery, and reduced complexity.

The primary advantage of using Mendix is that the database is maintained by itself. Mendix provides a wide range of components and can be easily included by just a click. The forms can be added quickly and insert new data into the database as well as update or delete existing data (Rodrigo 2012). Mendix can be appropriately used for prototyping because the entire application can be rebuilt very quickly to match an evolving requirement specification.

7.4.3 Mendix Features

There are three fundamental elements in the development of an application in Mendix, i.e., the Domain model, Microflow and pages (i.e. Forms):

- Domain model: is a data model that describes the information in the application domain at an abstract level.
- Microflow: allows the developer to express the logic of the application using a visual notation similar to flowcharts and Business Process Modelling Notation BPMN.
- Pages: define the end-user interface of the end application generated by Mendix.

7.4.3.1 Domain model

The domain model is a data model that represents the conceptual design of the application. The domain model is essentially used for developing and handling the database of the system. Similar to UML class diagrams, the domain model consists of entities and their relations represented by associations (see Figure 7-6). In Mendix designing and creating, a domain model is an essential phase in developing an application, and it can be generated effortlessly through drag and drop of the entities into the working area. Then attributes of entities are determined through the click on the entity where you can specify the entity name and add as many attributes as required by describing data types and constraints.

In the database, every entity is preserved in one separate table and has columns of attributes, with a unique identifier for the object. If an entity has specializations, there is also a column showing to which entity belongs.

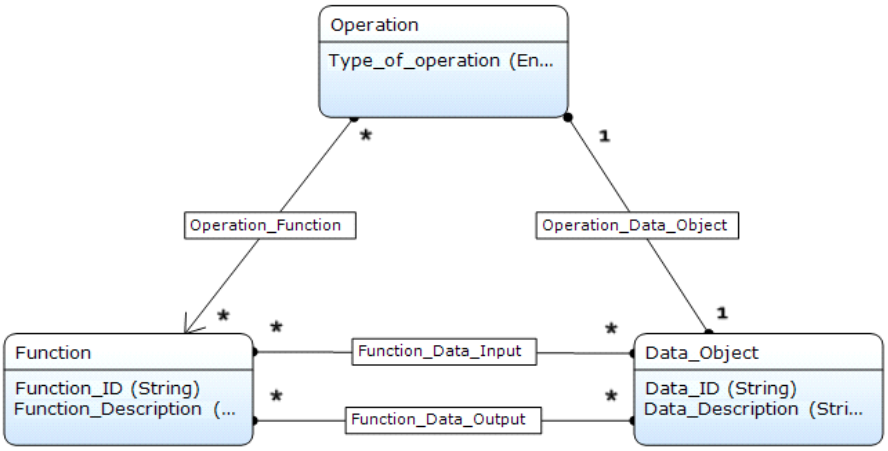


Figure 7-6 Mendix Domain Model (Example of function and data object)

7.4.3.2 Microflow

The unique feature presented by Mendix is the use of Microflow for creating and managing the data. Microflow is the graphic representation of the code that is applied to the desired actions and events. Microflow allows the developer to express the logic of the application. The graphical notation of Microflow is based on the Business Process Modelling Notation (BPMN). Additionally, during the development, everything is operated through Java code at the backend. It also provides all the essential features of a typical programming language like looping, start/end events, data retrieve, etc.

As showed in Figure 7-7 a Microflow is composed of elements. Below is a categorised overview of all elements. The following categories are used:

- Events represent start and end points of a Microflow and special operations in a loop.
- Flows form the connection between elements.
- Gateways deal with making choices and merging different paths again.

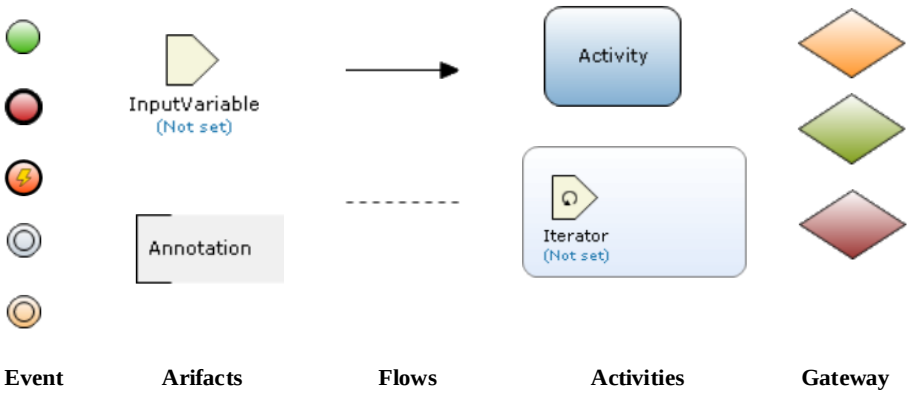


Figure 7-7 Microflow Elements

- A parameter is data that serves as input for the micro flow. Parameters are filled at the location from where the micro flow is triggered.
- Annotation provides the micro flow with input and allows comments to be made.
- Error handlers can be set on an activity, gateway or loop to define how to handle an error.
- Activities are the actions that are executed in a micro flow. A looped activity is used to iterate over a list of objects. For every object, the flow inside the looped activity is executed.

Figure 7-8 shows in the example of a simple Microflow that we used during the development of our impact analysis.

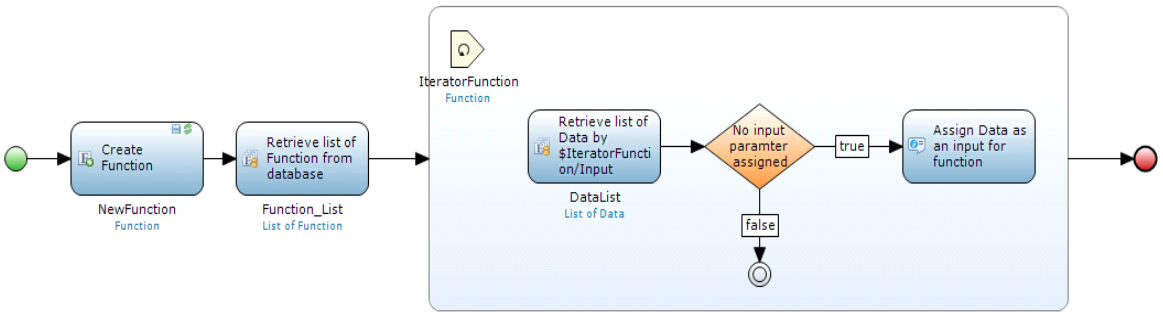


Figure 7-8 Mendix Microflow (Create Business Function)

In this example, Microflow is created to manage the adding a new function type of change in an ERP system. The Microflow adds new functions to the component list (i.e. function list) and checks all function in the loop that if there is any input has been assigned to that or not.

7.4.4 Mapping our Framework to Mendix

Mapping of elements between two different models is called a model transformation. This approach helps to clarify functionality and to provide a better view of the application. Model transformations are rules that transform models or transform data from one model to another. Model-to-model transformations convert information from one model or models to another model or set of models, typically where the flow of information is across abstraction boundaries (Beydeda, Book, and Gruhn 2005). This section explains two types of model transformations that we used during the development of impact analysis application, i.e. the transformation of the dependency meta-model to the Mendix domain model and the mapping of impact analysis mechanisms to microflow models (Milanović 2007). The first one refers to the transformation of the UML dependency model from chapter 3 to the Mendix domain model as a central architecture of Mendix application.

The example transformation of the ERP dependency model to Mendix domain model is shown in Figure 7-9. It includes two phases: Phase one covers the ERP system at the design time and the second part is related to the runtime components of the ERP system.

Note that not all the classes in the UML model are mapped in the domain model such as the types function (i.e. manual, internal, or external). Also, in the domain model, we defined other entities in order to demonstrate the associations correctly, e.g., using the node entity to identify the sequence of function in the process model. Appendix D provides a more in depth presentation of the domain model that presented in Figure 7-9.

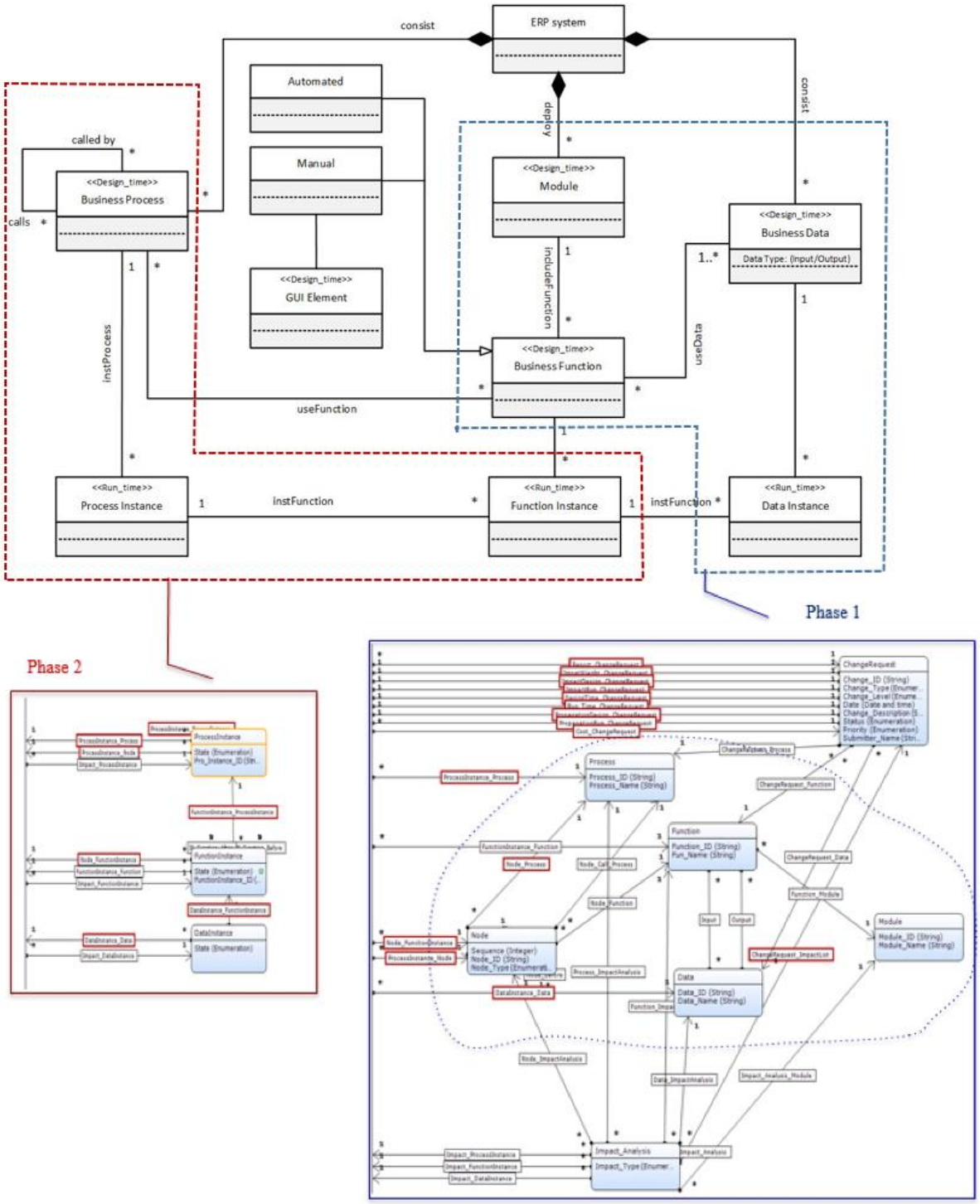


Figure 7-9 Mapping Process (Dependency model to Domain model)

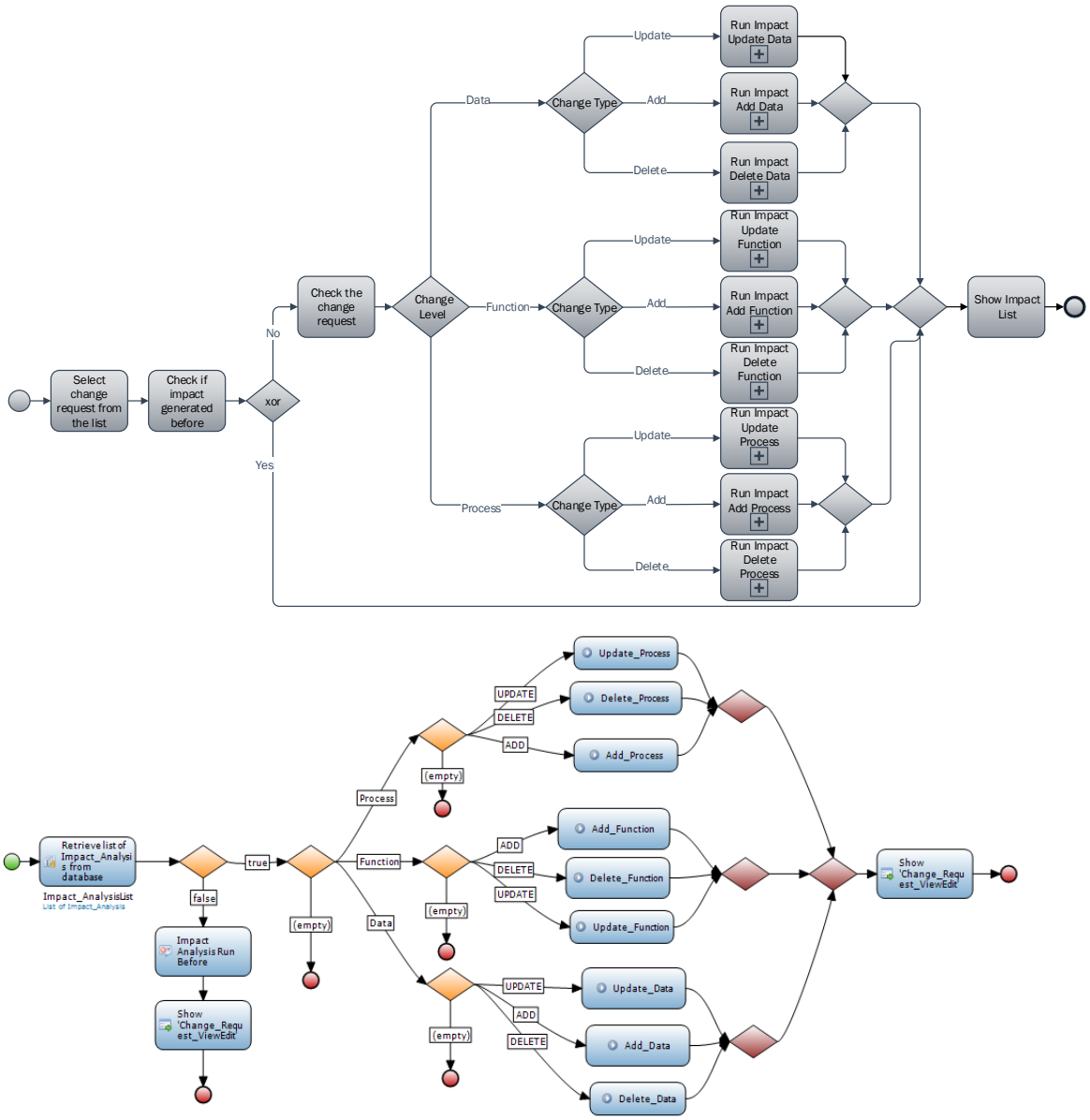


Figure 7-10 Mapping Process BPMN to Mendix Microflow (analysis of change request)

The second mapping process outlines the transformation of the impacts analysis mechanisms to Mendix microflow that represents the business logic of the impact analysis system. The impact analysis mechanisms have been first translated to intermediate BPPMN representations, which have then been translated into Mendix microflow. The transformation is specified as a direct mapping from BPMN to Mendix microflow elements as exemplified

in Figure 7-10. Figure 7-11 shows the microflow of impact analysis in case a change request requires updating a business process (i.e. the sub-process of Figure 7-10).

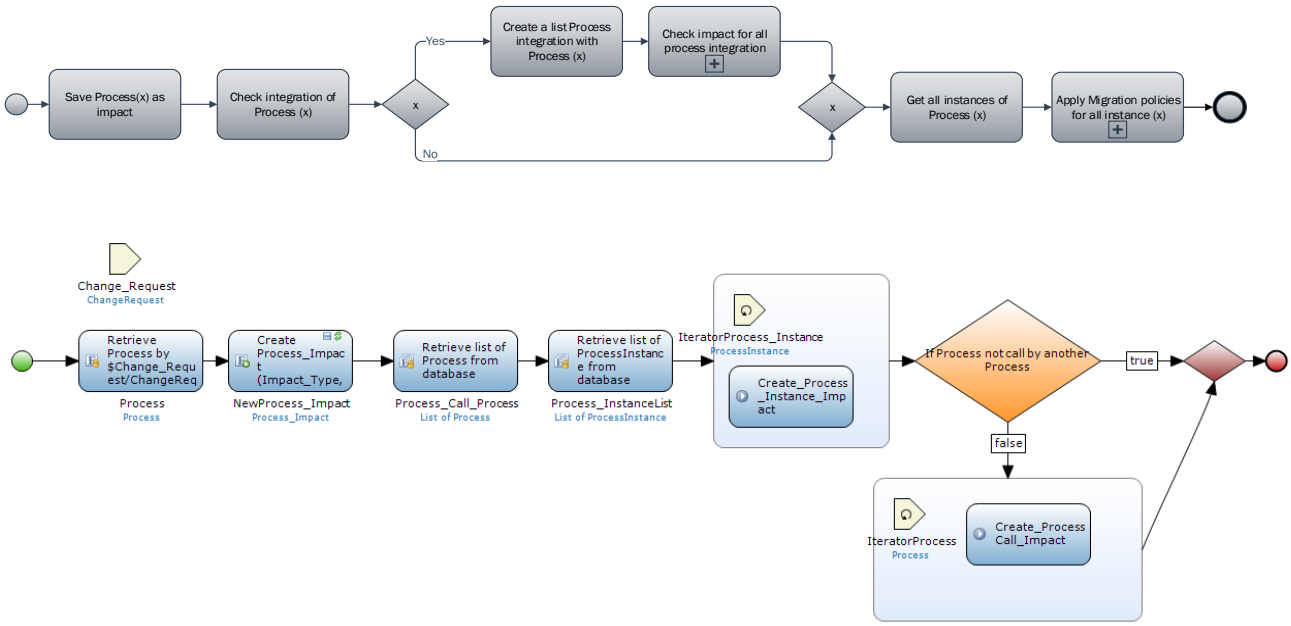


Figure 7-11 Mapping the BPMN for the algorithms 4 to Microflow

The example of process mapping is shown in Figure 7-11. It explains the updating process mechanisms of the algorithm 5 in Chapter 5. Here changes the request is used as a parameter. The Parameter is data that serves as input for microflow. The change request defines the process impact and the associated item. The microflow first considers the process from the change request as the initial object, which is affected by the modification. Then, the microflow checks the process integration with other processes that used this process as a subprocess during the execution. According to our impact analysis mechanism explained in Chapter 5, the microflow retrieves all the business process instances in the system, which requires further actions by other microflows through the loop. Further, this microflow represents the decision making on whether the process called by another process during the execution. If so, it is necessary to investigate the effects of process integration further through the execution of other microflow.

7.4.5 Implementation of Tool GUI

The Mendix forms are defined based on the entities from the domain model, such as ERP components, modules, process, functions and data and perform the action through the association between entities and microflow. The forms are to define the end user interface of impact analysis application. Every page defines in layout with widgets such as data view and the data grid. The data view shows the content of the single object whereas the data grid shows a list of objects from data based. The new/edit button shows a simple form that contains labels, textbox and buttons. Other buttons in a widget can associate with related microflow to perform an action. After defining all pages and form for our impact analysis then start testing a modification scenario of an ERP system to prove the feasibility of our approach.

7.5 Tool Demonstration: Running Example Walkthrough

This section provides a walkthrough with the example in order to demonstrate how the impact analysis works in our tool. A generic ERP system is considered for our demo that involves a list of business functions, data, processes, and business transactions (instances). This information will then be used for testing the tool to define the effects of modification for this particular scenario.

We consider a scenario of a manufacturing firm with production, purchasing, and sales activities. A list of modules, business process, functions/activities, and documents/data object is first specified and then mapped into our impact analysis tool. Figure 7-12 depicts an abstract view of the processes, modules, and functions of this particular ERP system.

7.5.1 Mapping the ERP Dependencies

This section represents the design specification and includes screenshots of some of the feature and functionalities mapped in our impact analysis tool. Figure 7-13 represents the list of items contained in the present ERP system. During the mapping, the user creates or edits the ERP components. This functionality is only available during the mapping time of the ERP components and once all the ERP components added in the impact analysis tool the edit or create an option is no longer available to the user.

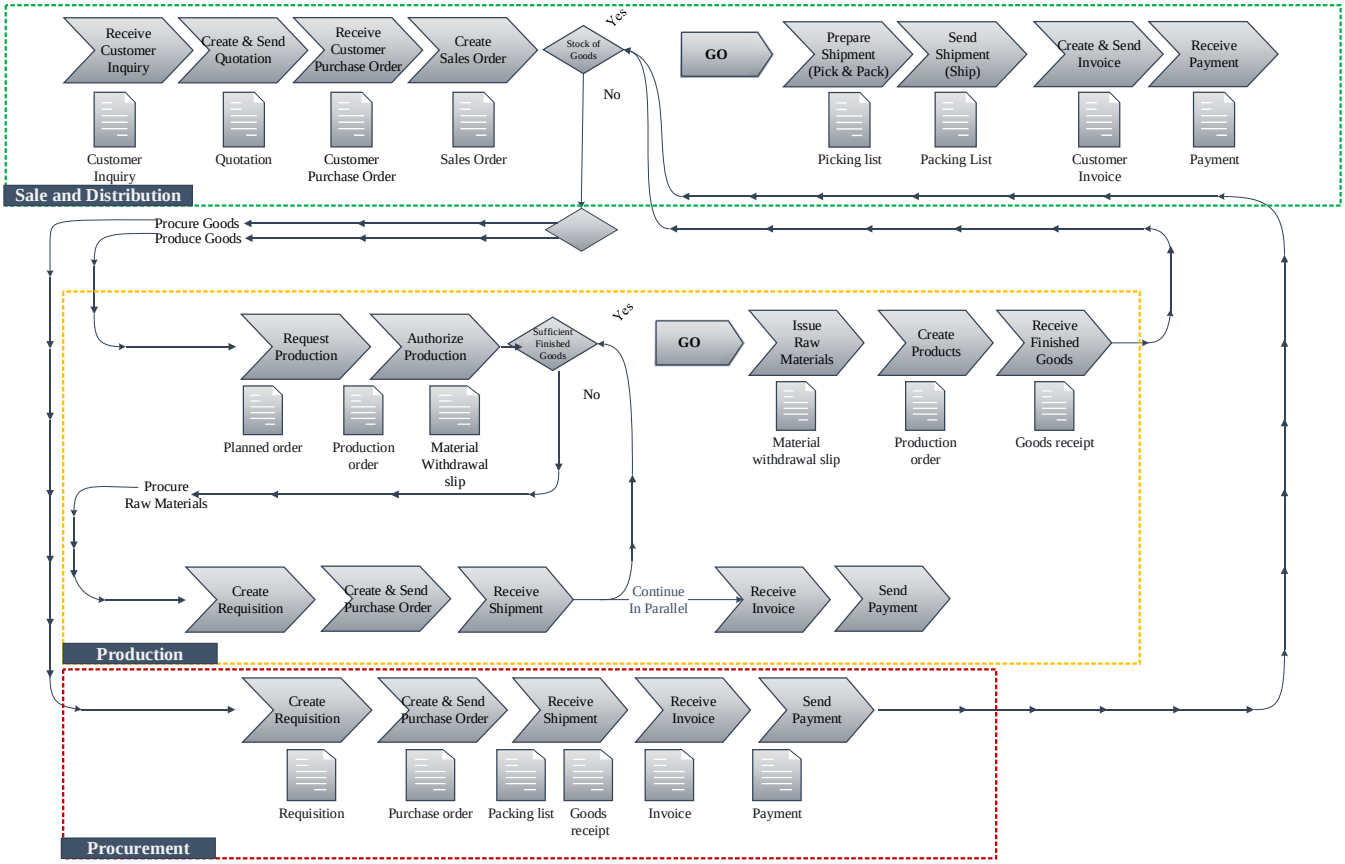


Figure 7-12 Example of ERP system

The initial part of the mapping starts with transferring all data objects of the ERP system. Figure 7-13-b represent the list of data objects included in this ERP system. The second part of the mapping process is to define a set of modules, and each module specifies the functionalities that only belong to that specific module (Figure 7-13-d). Every time a new module added to the system the business analyst has to create a new set of functions for that. As for each new function, it is essential to assign data to objects for input and output of the function. The final process of mapping is linked to business processes and the transaction (See Figure 7-13-e). The new business process model can include the execution of the functions of different modules. After specifying the process elements, the impact analysis can generate as many process instances.

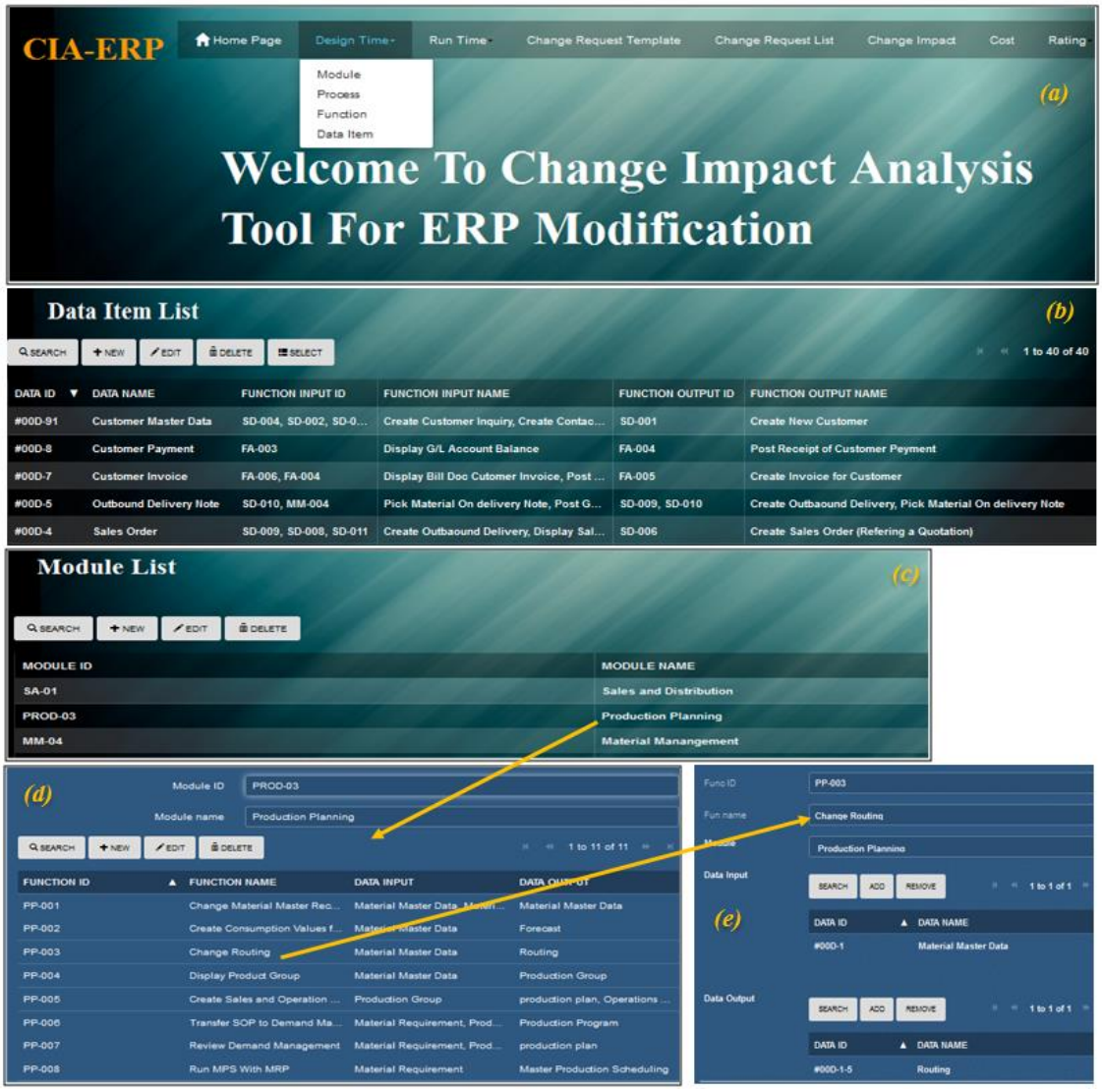


Figure 7-13 Example of mapping ERP System components

The process model as discussed in the previous chapter consists of sequences of functions. Figure 7-14-f shows the list of processes supported by this ERP system and in detail the functions and the sequence of operating then in Figure 7-14-g. Once business process model is defined for the system, then business analysis can generate and monitor the process instances progress by defining the state for each function instances included in the process instance in Figure 7-13-h.

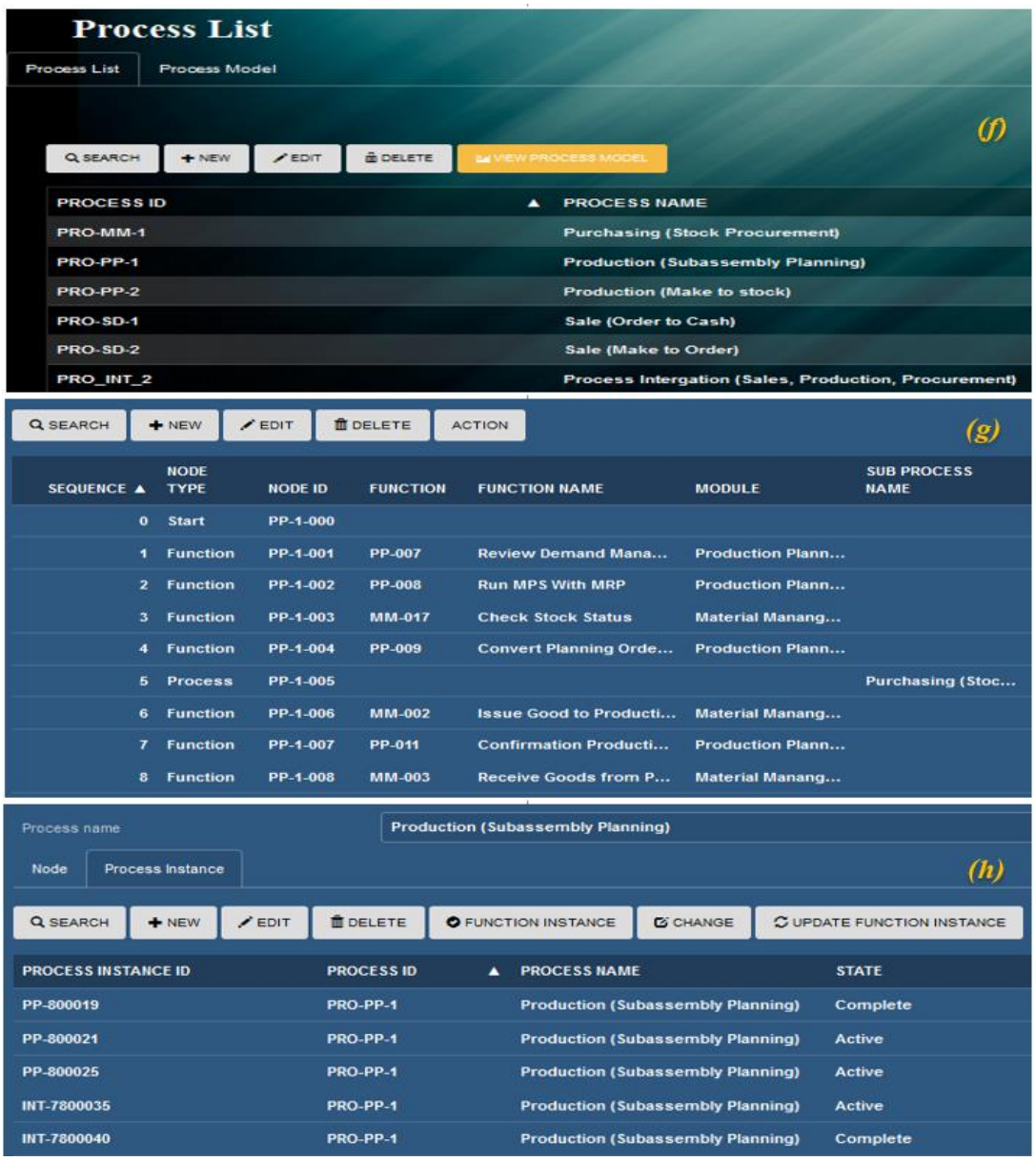


Figure 7-14 Example of Process Mapping of ERP system

7.5.2 Requirement Analysis

7.5.2.1 Change Log List

The request indicates as the repository of all modifications in the ERP system with the impacts and status, which specify the stage of the request during the modification process. This page illustrated in Figure 7-15 supports the following functionality: the analyst

can create a new change request; then run impact analysis to define the impacts; create a summary report of impact item types; compute the impacts according to the importance of modification items and finally assign the implementation strategies and calculate the cost. Once the change request is created then it is stored in the change request log list for further analysis.

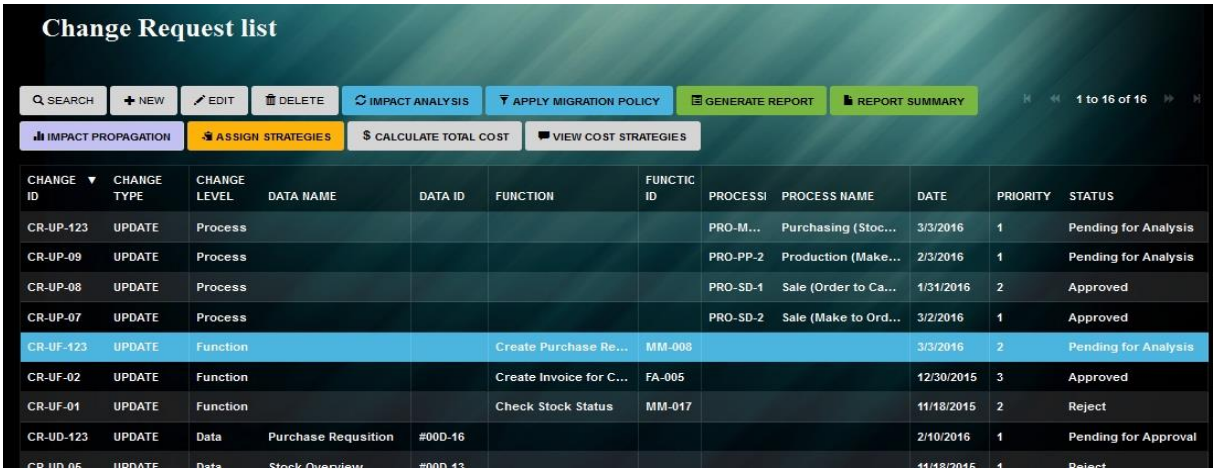


Figure 7-15 Change Request List

7.5.2.2 Change Request Template

As noted before in Chapter 4 and 5 to run the impact analysis the business analyst has to translate the end user requirement in our tool by addressing the modification criteria into the change request template to create the request. In this stage, the business analyst has to define which component is going to be changed and specifies the type and level of modification. Figure 7-16 demonstrates the example of purchase requisition modification from the previous chapter. In this example, the end user asked for improvement of creating a purchase requisition. Thus the analyst identifies the type of modification (update) and level of change (function) and then select from the list of function purchase requisition. In addition, they had to provide the description of the request and define the priority level based on the other dependencies in the system (If there is modification associated with this function as input and out).

Figure 7-16 Example of Change Request

7.5.3 Modification Analysis

7.5.3.1 Impact Analysis

The change request list the business analyst selects the new change request (update function for purchase requisition) and runs the impact analysis by clicking on the impact analysis bottom from the list. According to the change specification, the impact analysis computes the appropriate mechanism as explained in Chapter 5. Then the tool starts assessing the dependencies and classifies the impact items into different categories.

Figure 7-17 shows the example of updating purchase requisition function, which demonstrates the list of business processes that use this function following a list of active process instances. During the first iteration impact analysis captures all the instances (transaction) including completed one. Therefore, after the impact analysis shows the list of

all impacts in the first place, and then the analyst has to deduct those completed instances from the impact list and defines the migration policies for each active instance. This part of analysis outlines the notion of the critical point from the Chapter 5. Therefore, by clicking on the green button with the impact list, the result will be then updated the instances by removing the completed one and defines the action for those running instances in the system.

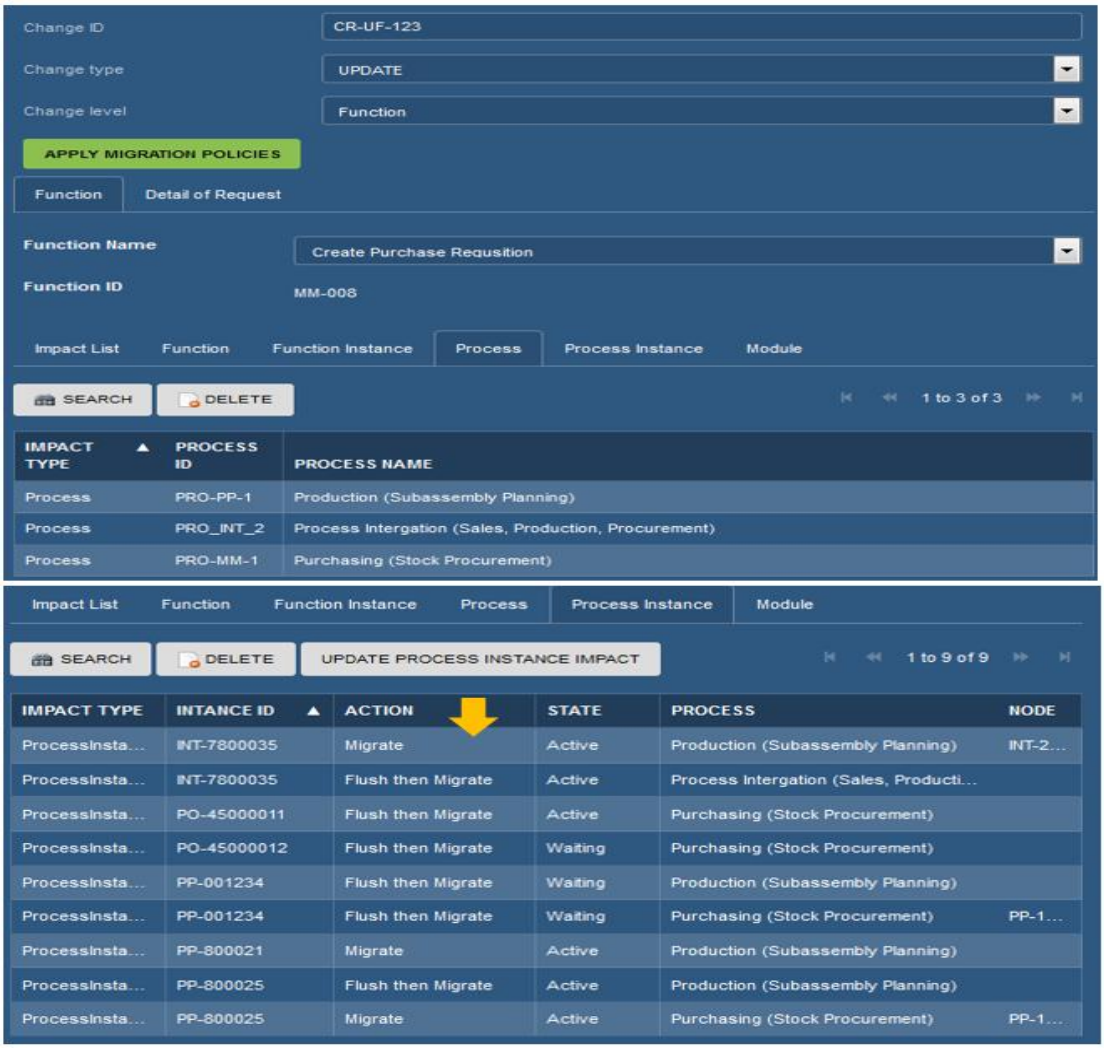


Figure 7-17 Example of impact analysis list

7.5.3.2 Impact Summary Report

The business analyst can view the impacts summary after impact analysis is completed capturing affected items in the ERP system. This reports showing the number of each impact items categories against the total number of item in the ERP system.

In addition, the report distinguishes the impact item at design time and the run-time. At the top page show the total number of items and the total number of the impacted items in the system. This report only gives the analyst the information about the number of items only. Later this result will assess further to measure the impacts based on the level of importance for impacted items. The example in Figure 7-18 shows the total number of items is 1879 and the modification impacted 21 including 4 at the design time and 13 item at the run time. The business analyst also can view the specific detail of each item category by clicking on the impact detail that shows the impact list items.

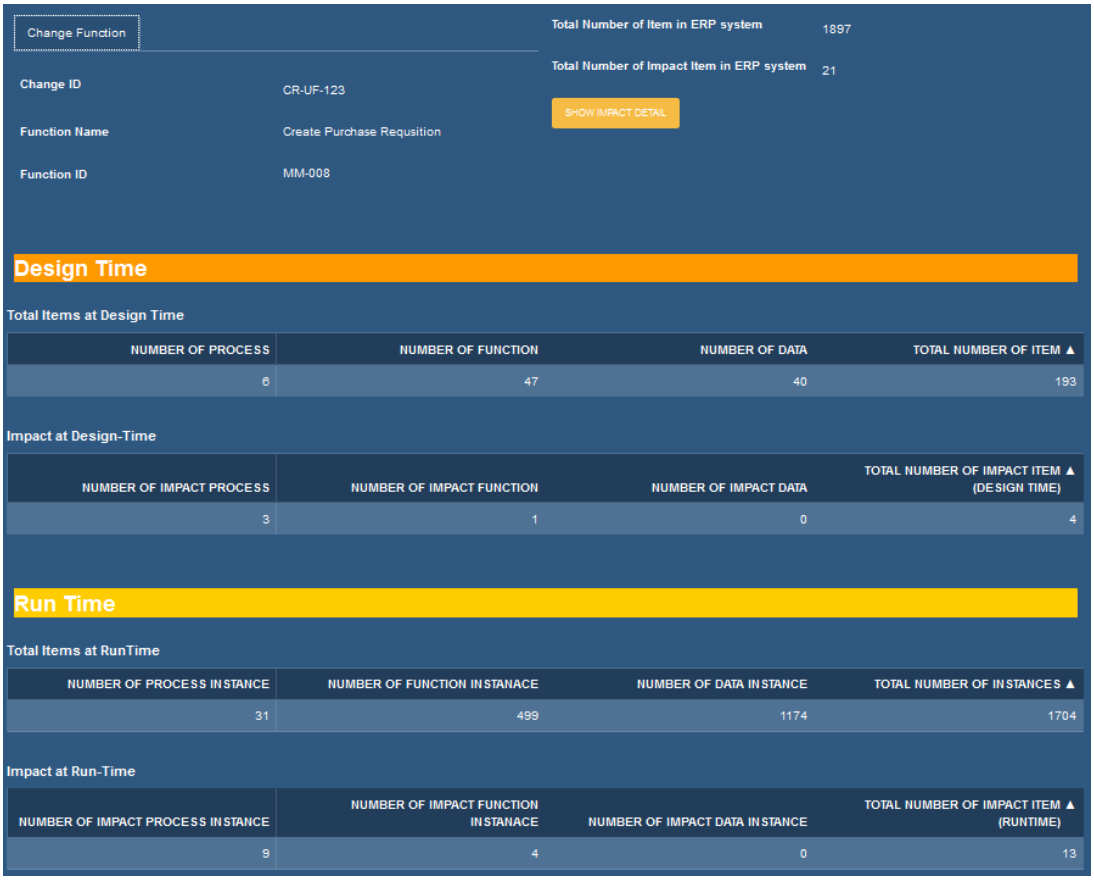


Figure 7-18 Example of Impact Summary Report

7.5.3.3 Impact Percentage

This part refers to Chapter 5 that explains the measurement of the metric. The aim here is to demonstrate the propagation of impact and predict the risk of change implementation. This prediction explains that the more complex a piece of ERP components

is more risk involves applying the modification. To obtain this type of assessment, the analyst again selects the change request from the request list and the click on impact propagation. Then the page will pop up include two parts. The first part/tab provides the scaling result that shows various types of scaling to present the result according to the percentage of impact. Figure 7-19 shows the example of this case for three levels of ERP component (data, function, and process) for all change requests is implemented in the impact analysis.

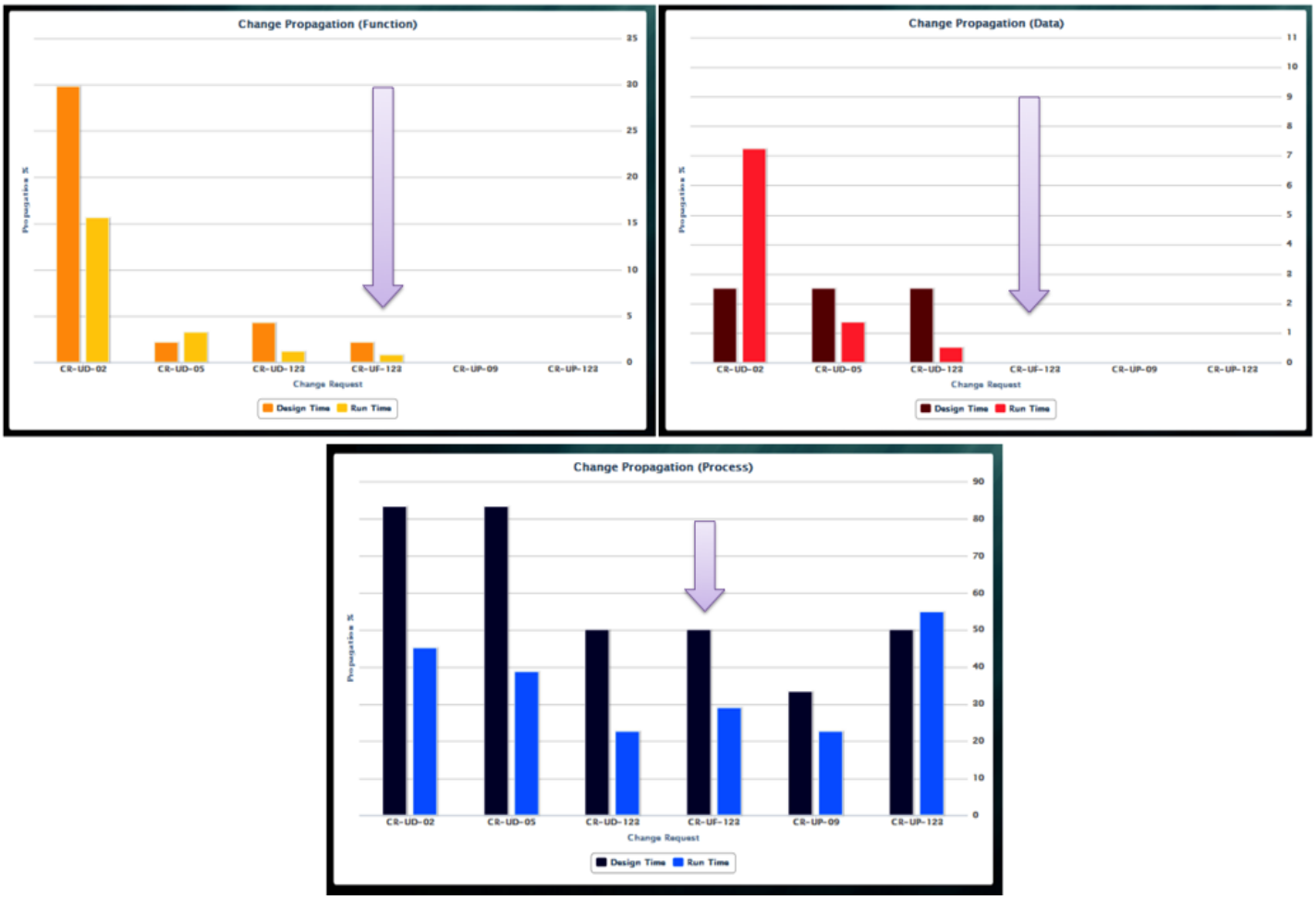


Figure 7-19 Example of Impact Assessment Part one (Percentage)

For each category distinguishes the impact result at the design time and run time. For instance, the example of updating function shows that there is no impact for the data object, and impact at the function level is relatively small. Whereas the impact of modification for

business process shows that 50% of business process affected at the design time and 30% of process instances at the runtime.

7.5.3.4 Impact Assessment

Figure 7-20 explains the second part/tab of the impact assessment in which for each impact set we assign impact weight as explained in Chapter 5 to calculate impact propagation. The weight signifies the importance of applying changes at each level and stage of the ERP system (design time or run time). Then according to the result defines the risk of applying changes to the ERP system. As such the business analyst can analyse and make a decision based on the impact result on whether to plan the implementation or the impact result is carrying a huge risk that the modification is not applicable to apply.

Figure 7-20 Example of impact assessment part two (propagation and risk)

This result of this visualisation allows the business analyst to implement a different solution for a particular requirement and evaluate the impacts by comparing two sets of change request together. For instance, to satisfy the requirement of the end user, there are

two possible solutions available to apply like updating the business process or update the business function and data objects. Therefore, the analyst can compare the two solutions according to the impact result and make decision.

7.5.3.5 Cost Analysis

This section allows planning of implementation according to the impact result. Since the business analyst and impact committees have extensive knowledge about the system configuration, then they can propose different ways of implementation for each impact set. Such that if the impact analysis detects three business functions as a result of the modification, then they can suggest different ways of applying a change to these functions through the configuration, code modification, bolt-on or other applicable solutions. This information can be retrieved through the discussion of with ERP developer and business analyst to identify a possible solution for each set of impact items implementation. Then the business analyst proposing various planning solutions like the example in Figure 7-21 illustrates a proposed planning solution for a particular change request (update create a purchase requisition function) according to the impact analysis result. After assigning the solution for each impact item, then the tool computes the cost based on a calculation that is discussed in Chapter 5.

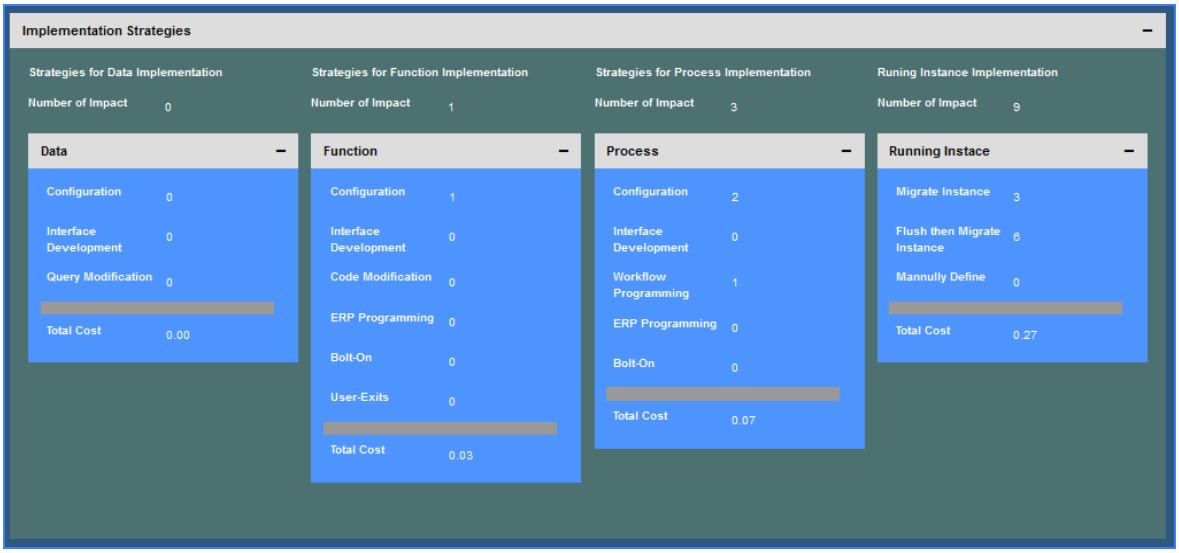


Figure 7-21 Example of Implementation Strategies

As noted the business analyst can propose as many solutions for one particular change request to compare and make a decision on the best strategies for implementation. This result shows in the Figure 7-22 shows a possible solution for one change request. The result shows that the solution PRO 1 is relatively expensive for implementation. The analyst can also compare the solution from together to define better implementation plan like solution PRO 2 and PRO 3 that result in PRO 4 as the most efficient way of implementation. The scaling of cost implication demonstrates the total number of the cost each item categories from 0 to 1 and the total cost of modification is measured from 0 to 4.

Figure 7-22 Example of Cost implication of modification

7.6 Summary

This chapter gives more insight into the design and implementation of the impact analysis tool as proof of concept. First, it presents an overview of the software development methodologies and explains. Then an agile development methodology and a model-driven approach were selected as the foundation of our implementation. After it provides a detail description of the features and functionality of impact analysis and discusses the design and implementation of the artefact development through adaptation of model-driven approach. The chapter concludes with a demonstration of a case in order to test and address functionality and feasibility of tool implementation.

CHAPTER 8 FRAMEWORK EVALUATION AND DISCUSSION

8.1 Introduction

This chapter is designed to provide empirical evidence to validate the achievement of our objectives. In software engineering research, when developing an IT artefact, it is essential to evaluate the artefact in order to ensure that it can operate in practice. Following the design and implementation of our impact analysis application from Chapters 4 and 5, this chapter aims to explore the viability of the discussed concepts and approach and play a formative role for further development. The fundamental question here is how to evaluate our approach and the designed impact analysis application. This concerns the explanation of the evaluation criteria, and the evaluation method in order to answer the above question. As such, this chapter is structured as follows. First Section, 8.2 presents our evaluation goal and the method and explains the evaluation phases of our assessment. Section 8.3 demonstrates the two case studies of ERP system and provides the result to perceive the feasibility of applying impact analysis tool. Further, in Section 8.4 discusses the two types of experimental design with ERP experts and non-expert users in change procedure to assess the applicability, functionality and usability criteria for impact analysis tool. Section 8.5 demonstrates the applicability and functionality result of the study with ERP experts and concludes this chapter on discussing the result of usability assessment with two groups of ERP experts and non-experts.

8.2 Evaluation Method

8.2.1 Evaluation Goal

For evaluation, we follow the principles of DSR (Design Science Research) that consists of building and evaluating artefacts, models, methods, or instantiations. March and Smith (1995) emphasises evaluation as one of the two activities in design science: build and evaluate. This thesis focused on the design of a framework for impact analysis of

modifications of ERP systems during post-implementation and developed a tool to assess and evaluate the change. The main purpose of evaluation is to show that our design solution has certain properties that work under certain condition and behave in a particular way.

To demonstrate the feasibility of our methodology, the artefacts for ERP post-implementation change management described in the previous section are implemented into a proof-of-concept decision support software tool. Many computing researchers do not evaluate whether the artefact does work in real-life context. Their objective is to show proof-of concept only via a functioning tool. However, sometimes the implementation will need to be compared with existing methods and approaches to demonstrate an improvement of the designed product. In this context, our main evaluation goal is to demonstrate that the tool and the change process were able to improve modification output for business analysts and developers to manage changes effectively in these systems compare to the existing approach.

Based on (Oates 2005, Hevner et al. 2004) work on many criteria for evaluating an IT artefact can be considered during the evaluation process, such as functionality, completeness, applicability, feasibility, consistency, accuracy and usability. For characterising design science research evaluation where the design artefact is a product, we can use a quality model such as ISO 9126 (ISO-9126 2010) as inspiration. ISO 9126 is an international standard for the evaluation of IT artefacts. ISO 9126 specifies 6 characteristics namely Functionality, Reliability, Usability, Efficiency, Maintainability and Portability and 21 sub-characteristics. Evaluation regards the development of criteria and the assessment of the artefact's performance in comparison to the criteria. The criteria we used depend on the reason we developed the impact analysis artefact which in this thesis is to support the business analyst and developer in the management i.e. specification, analysis, and assessment of ERP post-implementation change. The evaluation of our impact analysis tool should lead to the conclusion in the design process and as well as the design tool and may suggest that further improvement in either the approach or tool.

Based on the study of the above literature and with the regards to our main research objective we select the following criteria taken from ISO 9126 to evaluate our impact analysis artefact:

- *Functionality* is the capability of software to provide functions, which meet the

stated and implied needs of users under the specified conditions of usage.

- *Usability* is the capability of the software product to understand learned, used and attractive to the user when used under specified conditions.

Overall, our evaluation aims at answering the following questions:

1. Is the proposed change impact analysis tool feasible to apply and predict the effect of the change in any ERP system? (i.e. assessing the functionality)
2. How fit is the tool to support business analysts (EPR experts) requirements to manage the ERP post-implementation change process and assess its impact compares to the existing solution? (i.e. assessing the functionality with the ERP expert)
3. How easy is the proposed impact analysis tool for ERP user (such as ERP expert or non-expert in ERP change management) to operate the task and navigate through menus? (i.e. assessing the usability)

8.2.2 Evaluation Method

To evaluate the above criteria, we used a case study, interview and questionnaire as data collection method. Our evaluation is based on the following two phases:

- The first phase (Phase 1) investigates the feasibility of our impact analysis tool, and we evaluate to what extent our tool can handle standard functionality provided by commercial ERP packages. We decided to consider two pseudo-real organisations, i.e., GBI (Global Bike International) for SAP ERP 6.0 and Cronus for MS Dynamics NAV 2013 R2 by mapping all the functionality and feature and run various change scenarios. The dependencies among ERP systems components have been mapped manually based on the dependency meta-model represented in Chapter 5. Note that, although mapping manually all elements of an ERP system is very time consuming, this activity has to be executed only once after the go-live of the system and, in principle, can be partially automated using common APIs and programmatic interfaces of the ERP software that we did not have access to. For each system, we first have identified a set of standard features. Then, we have evaluated whether our tool can handle changes of the identified

features. This process has been executed following a customised Delphi method, whereby the list of ERP package features and the matching with the functionality in our framework have been identified first by the authors and then discussed with the research supervisor and ERP professional to achieve substantial agreement.

- The second phase (Phase 2) carries out the study design to evaluate the functionality and usability of impact analysis tool and approach with ERP users. The study is divided in two group of people into two parts as follows:
    - First Part (Part I) evaluates the impact analysis application functionality with a panel of 7 ERP professionals with average 10.7 years' experience with different types of ERP systems (SAP, Oracle/JD Edwards, and Microsoft Dynamics) in different industries (banking, higher education, ERP consulting). These experts have been chosen because of their experience on a range of different ERP software packages in different industries. The panel of experts covers both ERP business analysts and ERP developers who have grown into consulting roles during their career.
      Each session has involved a semi-structured initial interview and a hands-on session, with the tool, concluded by a follow-up interview. Sessions have lasted on average 2.3 hours. The objective of the initial semi-structured interview has been to understand the current practice at the expert's organisation regarding post-implementation change management. In the hands-on session we have demonstrated our methodology using a walkthrough our running example and asking the interviewee to run other change management scenarios. The purpose of the concluding follow-up interview has been to gather feedback from the expert on our methodology and its tool implementation. Further, the study is designed to evaluate the degree of applicability and functionality of the prototypical implementation, i.e., evaluate the fit for purpose in practice.
    - Second Part (Part II) this evaluation activity has involved the panel of 7 ERP professional and 12 master students in Business Systems Design who recently completed an introductory course on business process

management and enterprise systems. The objective of this evaluation is to investigate the usability and usefulness of our impact analysis design and implementation by asking the participant to run some change management scenarios and gathering feedback through answering a questionnaire.

This requires the evaluation of three claims:

- The information provided by impact analysis is useful for ERP practitioner to understand the effects of modification in ERP systems.
- The impact analysis tool and our approach can facilitate the practitioner to define impact more easily than before
- The impact analysis tool can be useful and easy to use for non-ERP expert

8.3 Evaluation Phase 1: Case Study Evaluation

In this section, we outline how our case study evaluation was conducted to thoroughly evaluate our approach in a real world context. The tool has been instantiated for the case of pseudo-real organisations, for demonstration purposes by two leading commercial ERP solutions i.e., GBI (Global Bike International) for SAP ERP 6.0 (Netweaver 2009) and Cronus for MS Dynamics NAV 2013 R2 (NAV 2016). The main reason for selecting these two instantiations is that both have evolved for several years as an example of ERP systems for training purposes and are now sufficiently complex to be comparable with ERP installations in real world contexts. The software tools and the documentation relating to the system specifications are freely accessible under the respective academic programs.

Before we elaborate each case study in this section, we first briefly summarise our evaluation process in this phase, which consists of the following steps:

1. Identifying the necessary components of each ERP instance and maps them to our impact analysis tool.
2. Designing change operation scenarios and implements them in our impact analysis tool.
3. Applying the impact analysis mechanisms to test our impact analysis framework.
4. Checking the correctness of the impact analysis result according to the dependency relationships.

5. Comparing the feature of our impact analysis then defining to what extent our tool is able to handle standard functionality provided by commercial ERP package.

In this section, for each system, we investigated on both of ERP systems and captured the business scenarios. Dependencies among ERP systems components have been mapped manually based on the dependency meta-model represented in Chapter 5. Then we designed and run modification scenarios of different type and obtained the impact result of the affected components. At the end of the assessment, we evaluated each ERP system features and functionality and compared them with our impact analysis result. The result should indicate that which set of standard features of ERP systems the impact analysis can support.

The next two sub-sections explain each case study and should prove the following statements:

- The impact analysis tool will find the same results that were determined manually
- The impact analysis tool has the capability to map the components of any ERP systems in order to present the dependencies

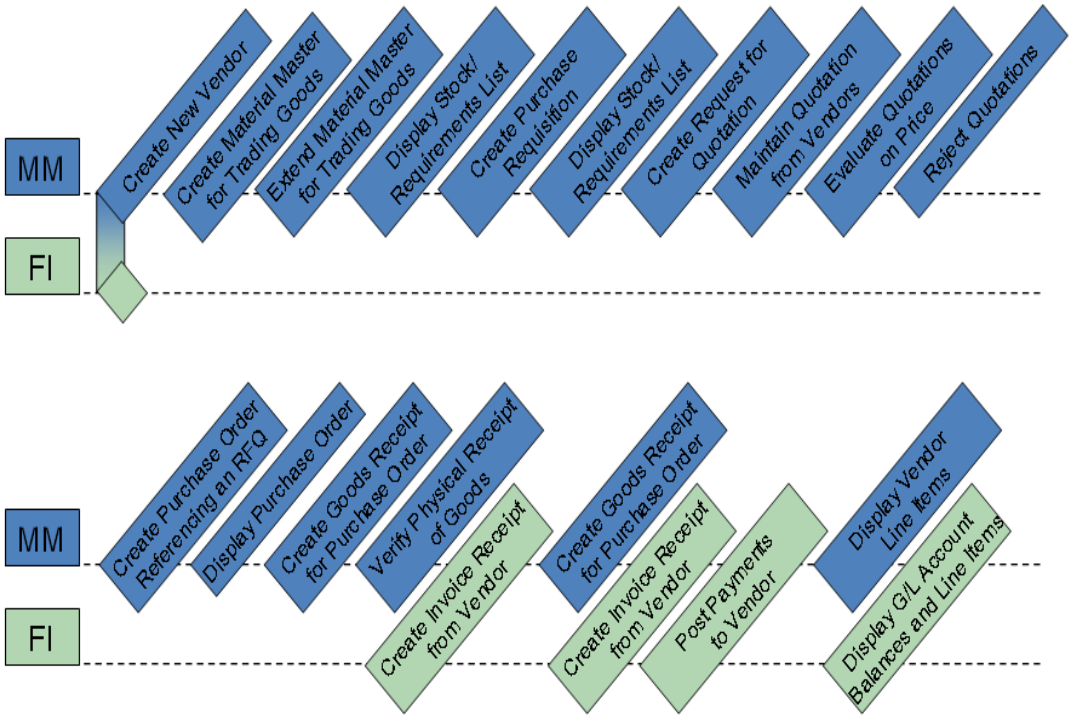
8.3.1 GBI SAP Case Study

Global Bike Inc. is a design and manufacturing firm of both off-road and racing bikes (Global Bike). Along with this, GBI implemented SAP ERP version 6.0 (Netweaver 2009) for the following division Material Management, Controlling, Production Planning, Sales and Distribution, and Finance. All ERP functions are centralised with the primary objectives to reduce costs for this firm. The GBI facility manufactures products for the US and international market. The warehouse manages product distribution for the retailers and individuals. GBI also sells accessories product line comprised of helmets, t-shirts and other riding accessories. GBI operates two production facilities, has three assembly lines and can produce around 1000 bikes per year. During the mapping process, we capture the ERP components items as depicted in below Table 8-1.

Table 8-1 Total item captured in GBI case study

Stage	Number of Processes	Number of Functions	Number of Data Items	Number of Modules
Design-time	6	47	40	5
Run-time (instances)	31	499	1175	
Total Item	*1902*			

GBI has outsourced the production of off-road and touring frames and wheels. GBI primarily assembles semi-finished goods into finished goods at its production facilities. Finished goods are either stored in the local warehouse or shipped to other regional distribution centres to fulfil customer orders. The graphic (Figure 8-1) below displays the complete process of production and presents the integration of material management with financial accounting department (20 tasks).



* MM= Material Management * FI= Financial Accounting

Figure 8-1 GBI Production Process

GBI starts the purchasing process by creating a new vendor and master record for a trading good in the system. After checking the stock (empty), the procurement process starts by creating a purchase requisition, generate a request for quotations and enter the quotations from various vendors including a new vendor. After evaluating and accepting the quotation of Supply GBI create a purchase order referencing the RFQ then post the goods receipt and verify the physical receipt in stock and finally post the payments to the vendor and review the G/L accounts. The Figure 8-2 below displays the complete process (17 tasks).

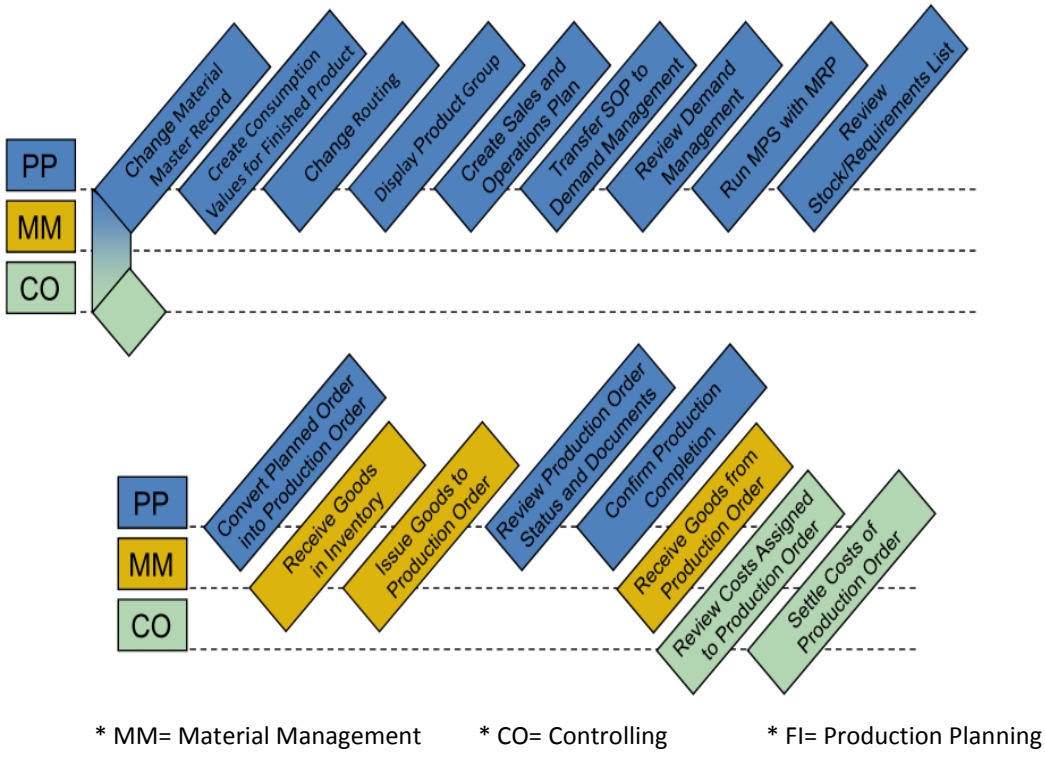
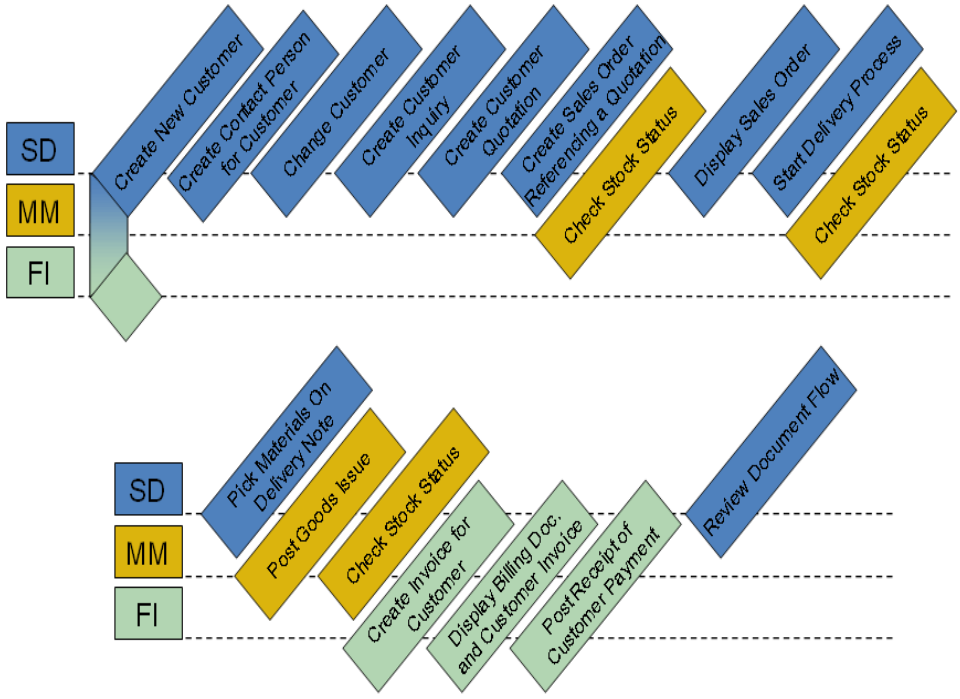


Figure 8-2 GBI Purchasing Process

GBI sells its bikes exclusively through Bicycle Dealers (IBDs). An order-to-cash process is taken on a different department of Sales and Distribution (SD), Materials Management and the Financial Accounting (FI) departments. The sales order process creates inquiry, which then processes into a quotation. Once the customer accepts, the quotation then creates a sales order referencing the quotation. When the stock has enough bikes, the GBI

delivers the products sold to the customer, and create an invoice then receive the payment. The Figure 8-3 displays the complete process (17 tasks).

After capturing all the information relating to the mapping process we started specifying items for each ERP components (i.e. data, function, process, and module) and defined the dependency relation between them in our impact analysis. Figure 8-4 depicted the example of mapping business process and functionality for production process same as the process in Figure 8-1 of GBI case study in our impact analysis.



* MM= Material Management * SD= Sales and Distribution * FI= Production Planning

Figure 8-3 GBI Sales and Distribution Process

We implemented various change scenarios in impact analysis then assessed the correctness of the results manually to make sure that the tool is captured the impact items correctly. Figure 8-4 and 8-5 shows the result of impact analysis assessment for different type and level of modification. The first screen shot represents the mapping items of GBI and the second screen presents the result of running various change scenarios to define the impacts.

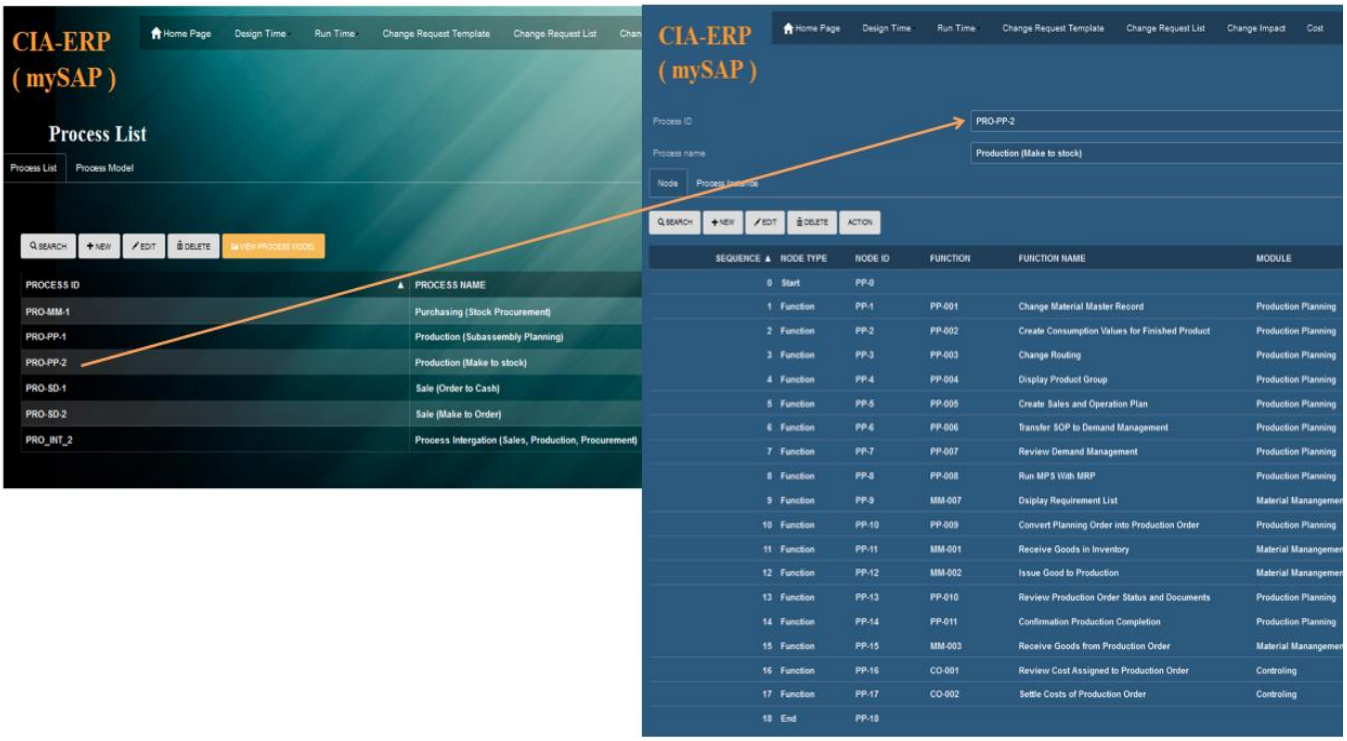


Figure 8-4 Screenshot of mapping items

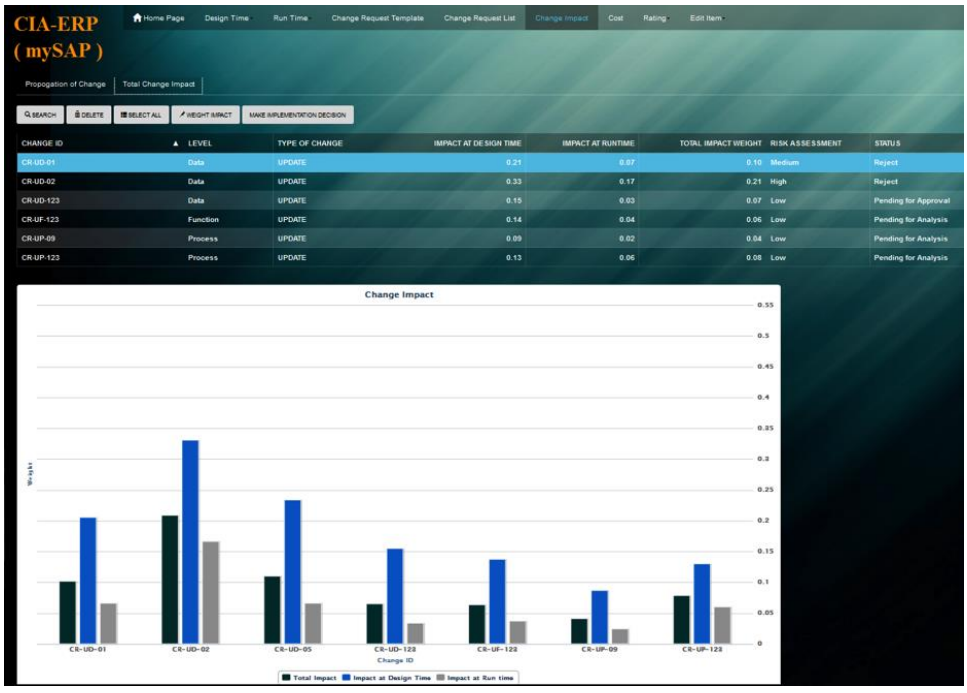


Figure 8-5 Implementation of Change Scenario

8.3.1.1 Result of Evaluation for GBI Case Study

Table 8-2 shows, the results of coverage analysis for SAP ERP system. We first have identified a set of standard features. Then, we have evaluated to what extent our tool is able to handle changes of the identified features for SAP ERP system. The result will be discussed at the end of this section.

Table 8-2 Comparison of impact analysis tool vs. ERP functionality of SAP

No	Enterprise Level	SAP Enterprise Level Features and Functionality	Impact Analysis Features and Functionality	Detail
A	Organisational/ User Layer	Authorization / Roles and Responsibilities	N/A	Out of the scope of analysis
		Organisational Level (Client, Company Code, Plant, etc.)	N/A	Out of the scope of Analysis
B	Presentation Layer	Graphical User Interface Transaction Data Header/Item	N/A	Out of the Scope of analysis,
C	Business Logic Layer	Module	✓	Impact analysis can identify which modules are affected by change as a result of modifications of functions constituting a module.
		Business Processes	✓	Impact analysis can identify which business processes are affected by change and can handle sub-process integration.
		Business Process Integration	✓	
		Business Function/Transaction	✓	Impact analysis can identify which business transactions are affected by change.
D	Application Layer	Master Data	✓	Impact analysis can identify the impact of change of master data as business objects
		Organisational Data	N/A	Out of the Scope of analysis
		Transaction Data	N/A	Out of the scope of analysis (post-implementation changes do not deal with ad-hoc changes to running process instances, which manipulate transaction data)
		Attribute of Data Item	N/A	Impact analysis cannot differentiate the change impact of individual attributes of a business object.
E	Operation Layer	Active Business Process	✓	Impact analysis can identify which process instances are affected by change.
		Active Business Transaction	✓	Business transactions affected by change can be identified by the impact analysis based on the process instances affected by change.
		Active Transaction Data	✓	Transaction data affected by change can be identified by the impact analysis based on the process instances affected by change.

8.3.2 CRONUS Microsoft Dynamic NAV

CRONUS International Ltd (NAV 2016) is a fictitious company with business scenarios, employees, and products. As we explored Microsoft Dynamics NAV documentation, the CRONUS data are used as examples to understand the implementation of Microsoft Dynamics NAV (NAV 2016). The company develops, markets, and sells many items to the end users and has a broad customer base. The items that the company handles fall generally into the following categories:

- Modular office furniture, for which the company produces some components and purchases others
- Bicycles, for which the company does final assembly parts manufactured elsewhere and in-house.
- Computer hardware, which the company manufactures, distributes, and services.

Microsoft Dynamics NAV supports all typical tasks and information that CRONUS needs to manage sales and receivables, and purchase and payables process as the two essential business areas. This case study introduced the basic sales and purchase concepts by presenting the complete sales and purchase process flows and the integrations to other application areas.

Table 8-3 is depicted the summary of items that we captured during the mapping process. The case study demonstrated the end-to-end business processes and the standard functionality of Microsoft Dynamic NAV as an ERP system. The diagrams in Figure 8-6 and Figure 8-7 present an overview of a purchase and sales process utilised in this case study.

Table 8-3 Total items captured in CRONUS case study

Stage	Number of Processes	Number of Functions	Number of Data Items	Number of Modules
Design-time	3	22	22	3
Run-time (instances)	11	114	222	
Total Item	*433*			

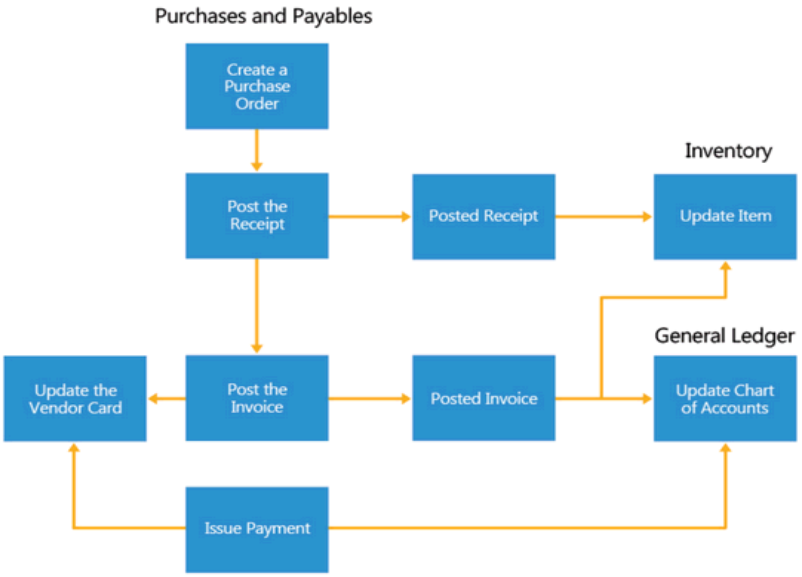


Figure 8-6 CRONUS Purchase and payable process

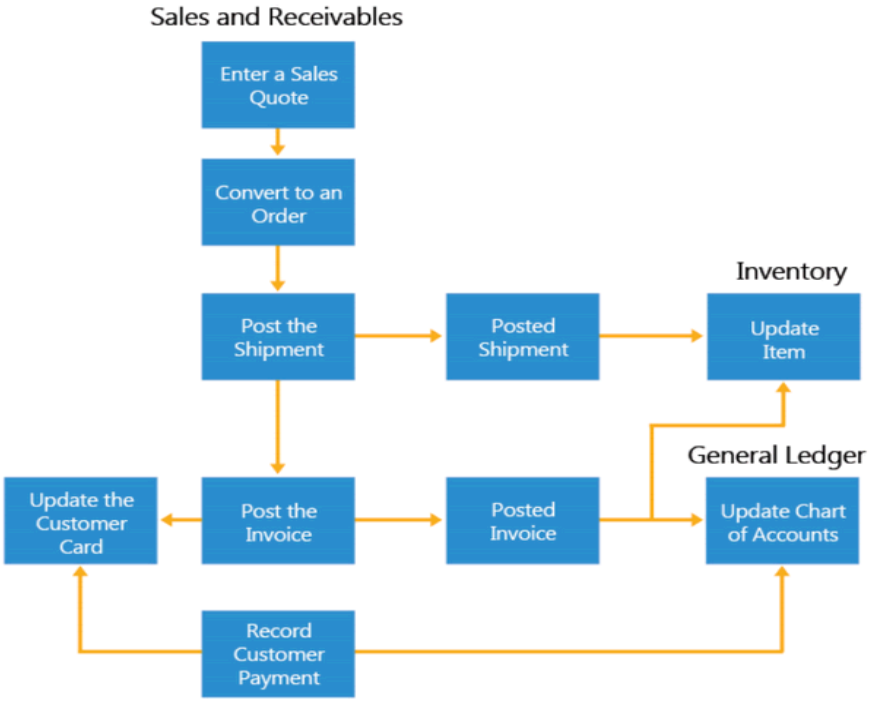


Figure 8-7 CRONUS Sale and Receivable

After capturing all information for the mapping process, we start configuring the impact analysis for this particular case study. We define all the elements of data objects, business function, and business processes and create a different sample of a business case in

our impact analysis. Figure 8-8 depicted the example of mapping business process and functionality for purchasing process of CRONUS case study in our impact analysis.

We implemented various change scenarios of a different kind in the impact analysis and then compared the result of the tool with the result that we assessed manually in order to make sure that impact analysis correctly defines the impact items. Figure 7-9 shows the result of impact analysis assessment for different type and level of modification.

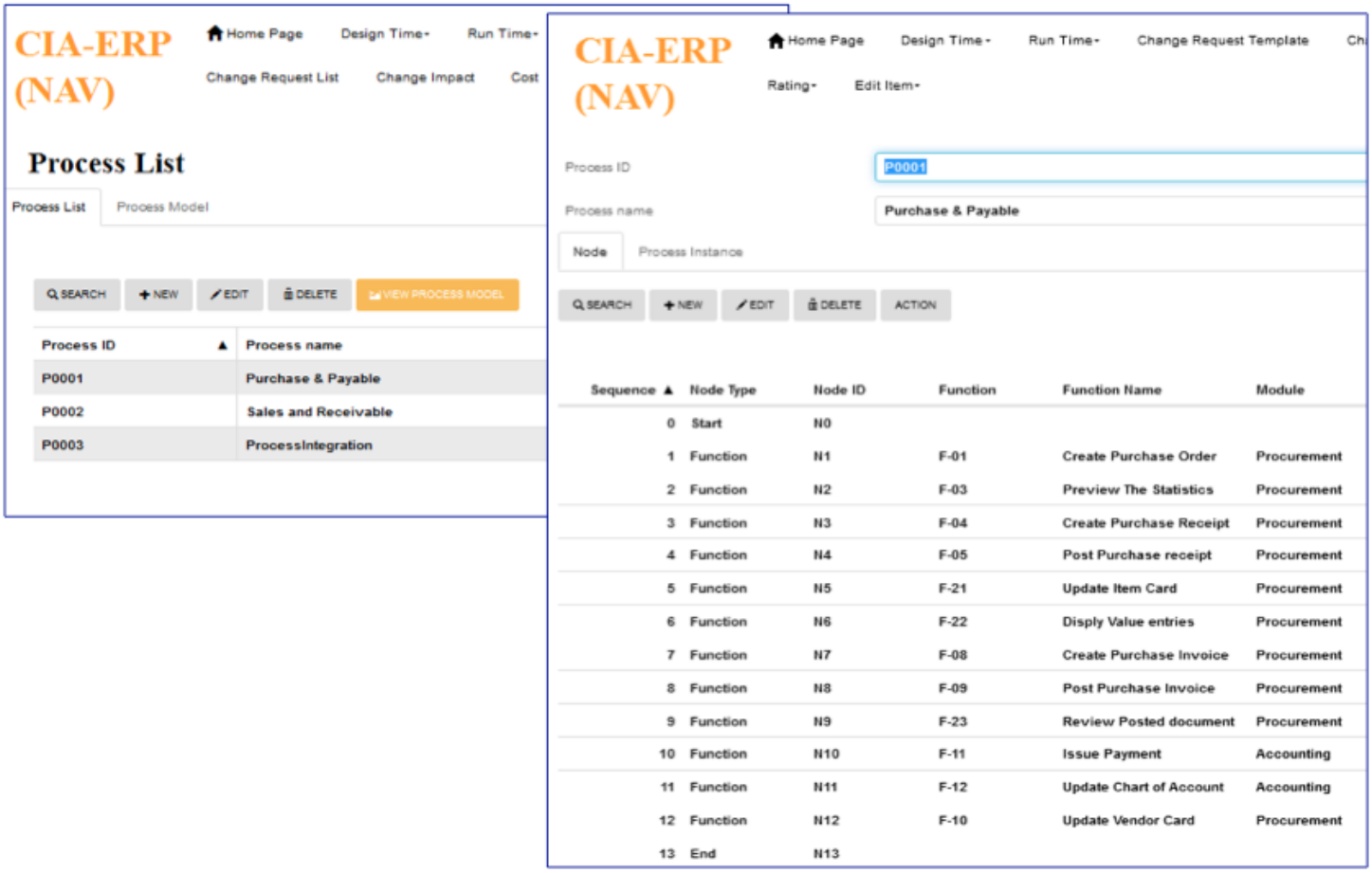


Figure 8-8 Example of Mapping Business Process of CRONUS

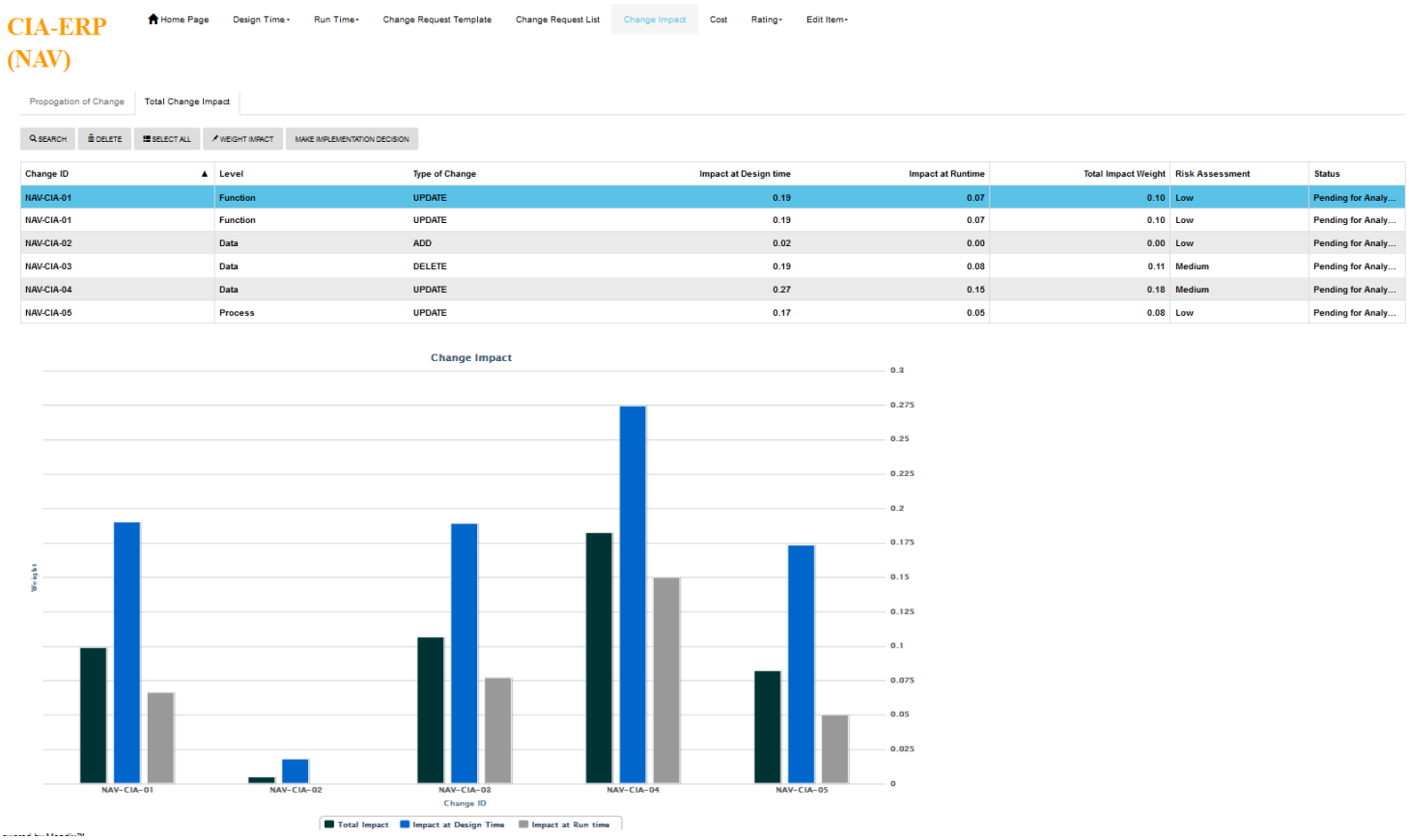


Figure 8-9 Example of Change to CRONUS and the Result by Impact Analysis

8.3.2.1 Result of Evolution for CRONUS Microsoft NAV

Compared to the SAP case study this study has fewer features and functionality in order to assess the correctness of impact. Same as previous case study, after validation of the impact result we start comparing the functionality and features of Microsoft Dynamic NAV. We have specified in table 8-4 a list of standard features of Microsoft NAV and define the categories that our impact analysis can support as impact items. Next section discusses both case study in further detail.

Table 8-4 Comparison of impact analysis tool vs. ERP functionality of Microsoft Dynamic

No	Enterprise Level	SAP Enterprise Level Features and Functionality	Impact Analysis Features and Functionality	Detail
A	Organisational/ User Layer	Authorization / Roles and Responsibilities	N/A	Out of the scope of analysis
B	Presentation Layer	Graphical User Interface Fact Box/Sorting/Navigation/Field	N/A	Out of the scope of analysis
C	Business Logic Layer	Department	✓	Impact analysis can identify which department are affected by change as a result of modifications of functions belonging to a Department.
		Business Processes	✓	Impact analysis can identify which business processes are affected by change and can handle sub-process integration.
		Business Process Integration	✓	
		Business Function/Action	✓	Impact analysis can identify which business actions are affected by change.
D	Application Layer	Master Data: - Chart of Account - Cards (Vendor, Customer, Item) -Orders (sales, Production, Purchase)	✓	Impact analysis can identify the impact of change on master data as business objects
		Organisational Data	N/A	Out of the Scope of analysis
		Journals (General, Sales, Purchase, Item)	N/A	Impact analysis cannot specify what journal will be impacted by modification of data.
		Attribute of Card	N/A	Impact analysis cannot differentiate the change impact of individual attributes of a business object (or "card").
E	Operation Layer	Active Business Process	✓	Impact analysis can identify which process instances are affected by change.
		Active Business Transaction	✓	Business transactions affected by change can be identified by the Impact analysis based on the process instances affected by change.
		Active Transaction Data	✓	Transaction data affected by change can be identified by the impact analysis based on the process instances affected by change.
		Posted Documents	✓	Impact analysis can identify posted documents affected by change, since "posted" is only a particular state of a business object.

8.3.3 Discussion for Phase One Evaluation:

During the analysis of the result of both case study, we first have identified a set of standard features for both ERP systems. Then, we have evaluated whether our tool can handle changes in the identified features for each particular ERP system. This process has been executed following a customised Delphi method, whereby the list of ERP package features and the matching with the functionality in our framework have been identified first by the authors and then discussed with the panel of ERP professionals including the research supervisor and the ERP change management expert to achieve substantial agreement.

From the analysis and discussion, we found out the impact analysis is not able to address a change in the presentation layer or at the level or roles and responsibilities in the organisational layer. We argue that change in the presentation layer always present when the core functionality of an ERP system. It is related to issues of usability and acceptance, but it does not affect the running operations of an organisation. The impact of change on organisational governance, i.e., roles and responsibilities, is subject to our ongoing work.

From the result of both case study evaluations we can outline the following conclusion:

The impact analysis tool can support the following statements:

- The capability to map dependency of the complex ERP system
- The applicability to configure and assess the impact of any ERP system
- The impact analysis mechanism can accurately define the impact same as the manual checking.

The impact analysis tool cannot support the following statement:

- It is not possible to map all the components of the ERP system in our tool. Some of the components (e.g., data artefacts) like a journal in Microsoft Dynamic NAV or organisational data in SAP that only used in these particular ERP systems. This required changing and configuring our dependency model according to their specification to capture their impact precisely. Meaning to define the impact of these particular features the modification of dependency meta model is required plus the modification of impact analysis mechanism. However, these features only support a

particular ERP system, and it is not possible to run the same mechanisms for all ERP system.

- Clearly, from both case, the impact analysis cannot support which GUI and which wizard can be impacted by modification since this was not the scope of our implementation and analysis.
- The impact analysis cannot define what part of the data object will be impacted. Our tool can only define the data object like purchase requisition document is impacted but cannot differentiate the change impact of individual attributes of a business object.
- Organisational elements like division and responsibility in both ERP systems cannot be captured by impact analysis, as this part is not included in the scope of this research. For the impact analysis can capture these items it is essential to extend our dependency model to map the related item following the extension of our impact mechanism to be able to capture the impact item in this category.

8.4 Evaluation Phase 2: User Study

The aim of this section is to assess the applicability, functionality and usability of our approach for defining the impact of changes in ERP systems in practice. We conducted two empirical studies to assess our evaluation criteria with two categories of participant.

The first study involves ERP experts (i.e., business analysts) with change management and ERP experience in the public and private sector. This study enables us to assess the applicability, functionality and usability of our approach and tool.

The second study involves a group of master students without real life experience with ERP, but with a technical background on ERP obtained by successfully completing a course of business process management and ERP systems in their master course. This second group of users of our tool allows us to investigate the opinion of a non-expert in ERP change management to test the impact analysis application usability only. This diversity among the test groups is expected to produce a more balanced view of the usability and usefulness of our impact analysis tool. The detail associated with each group of the study is explained in the next two sections. The sampling size for both groups was calculated through the study of

literature for usability evolution by (Faulkner 2003) (Hwang and Salvendy 2010) that both indicate the sampling size between 5 to 10 people as experts and novices.

8.4.1 ERP Expert Study (Part I)

The main purpose of this study is to investigate the applicability, functionality and usefulness of our impact analysis tool and approach. A group of 7 participants was invited to take part in this study. The test group consists of ERP experts with change management background (e.g. business analyst, developer, project manager) with average 10.7 years' experience with different types of ERP systems (SAP, Oracle/JD Edwards, and Microsoft Dynamics). The participants in this group are recruited through LinkedIn group communities (i.e. ERP and SAP community). These experts have been chosen because of their experience in a range of different ERP software packages in different industries.

The study consists of three stages: semi-structured interview about the current approach and available techniques for assessing change modification in ERP system, evaluation of our impact analysis functionality through the demonstration of the tool, and finally evaluating the usability of our impact analysis by applying pre-defined exercises with the tool.

Sessions have lasted on average 2.3 hours. The objective of the initial semi-structured interview has been to understand the current practice at the expert's organisation regarding post-implementation change management. In the interview session, all participants were interviewed about the experience of using an impact analysis and the approach that they used to manage the changes in the ERP system. The reason doing the interview was to collect as much as information from the participant to be able to form a difference and distinguish between our approach and the existing one that ERP specialist used. These interviews were scheduled in advance. They were organised around a set of predetermined open-ended questions, with additional questions emerging from the dialogue (See Appendix C-part A).

The key interview questions are as follow:

- To investigate if any of the ERP professionals have experienced using impact analysis tool during the change management process which is particularly used

for the ERP system. We defined similar tool such as SAP solution manager or PanayaIA that both have served in practice to identify some degree of impacts. During the interview, we asked from each participant if they experience using one of the following tools. From this question, we can identify any new impact analysis tool that we have not acknowledged that as an impact analysis tool and also how useful the existing tool such as solution manager for ERP expert during the change management process.

- To investigate on how the ERP professional assesses the impact of the change in ERP system when the new requirement emerges from the organisation and how they evaluate and analyse the impact of change (i.e. using a customised tool, expert judgment or any other techniques). The purpose of raising this question is to identify how convenient it is for the ERP specialist to determine the impact of change.
- To investigate if the experts use a standard set of activities or process to trace the impact of modification in their ERP system. From the result of this question, we compare our change management procedure capability with the existing approaches in practice.
- To define what resources are available during the change process to assess the impact so we asked the experts to determine if they were using any resources such as dependency model, change historical data or any change documentation during the assessment of ERP modification.
- The final question during the interview session is to define to what extent it is important for the ERP expert to utilise a structured impact analysis approach when evaluating the impact of a change in ERP systems.

This methodology allowed the identification of Behaviours during the time, which would be difficult to capture through informal conversations, narrowing the focus of the discussion into more specific issues. The interviews took, on average 30 minutes and audio-recorded then transcribed later by the researcher. This method served as a source of validation and refinement of the approaches developed during the research process.

In the next session, in our study, we provided a demonstration of the impact analysis tool. In this exercise, we explained the functionality and features and the process of how to use the impact analysis during the time of ERP modification. Questionnaires were perceived using a Likert scale and open questions to gather additional qualitative data from the participants. In the questionnaires, we asked eight questions about the functionality that our impact analysis offer and the impact result from the demonstration of a change scenario example. The questionnaire items (See Appendix C part A) also are taken from the research study of change impact analysis functionality and accuracy (Abgaza, Javeda, and Pahla 2013). The questionnaire items assess the accuracy of impact analysis and the capability of the tool to capture the impact item such as identification of impact set for each ERP components category correctly. Also, the questionnaire investigates how the impact analysis tool enhances the decision making to identify optimal strategies for change implementation using the estimates related to the cost and impact. The result of the questionnaire was evaluated, and the average responses of the individuals were analysed.

Finally, in the last session, we asked the ERP expert to validate the usability of our impact analysis tool by implementing change scenario and evaluate the result of their action. In this part, there were two exercises targeted at examining the experiencing of using impact analysis tool. The first one asked the participant to implement two sets of change scenario and evaluate the result. The second exercise asked the participants to propose, and assigned implementation strategies based on the assumption. Based on the result of running change operation in impact analysis participant can propose different ways of implementing change by selection among the implementation strategies for each level of changes. Our assumption here is that the participant has the knowledge on how to implement the impacted items captured by impact analysis and also all the implementation strategies are feasible to apply.

The purpose of this session is to validate our tool and our approach based on usefulness, ease of use, ease of learning, and satisfaction. In this stage of our study, a set of questionnaires was perceived that associated for each part of the assessment using Likert scaling and open questions to gather additional qualitative data from the ERP expert (Lund 2001). The items of this questionnaire relate to the typical dimensions of technology acceptance and successful evaluation, i.e., system usefulness, ease of use, ease of learning

and satisfaction with the system. All items are evaluated on a 5 point Likert scale from strongly disagree (ISO-9126) to strongly agree (5) (see Appendix C).

8.4.2 Non-expert Study (Part II)

In our evaluation Usability is the degree to which our impact analysis tool can be used by ERP business analyst to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use. To get an understanding of how well our impact analysis tool works we also test the usability with non-expert in ERP change management, but having a background in business process management and enterprise systems. This diversity among the test groups between expert and none expert in ERP change management background is expected to produce a more balanced view of the usability and usefulness of our impact analysis tool as both groups have experienced with the ERP system. A group of 12 participants was invited to take part in our study. The participants from this group were selected former students of City University who took the ERP modules and recently completed an introductory course on business process management and enterprise systems. We recruited the student via email and LinkedIn to take part in our study. The reason for selecting among city university students is that they experienced both the ERP system and business process Modelling and they understand the domain problem easily compares to any other end-user.

The objective of this evaluation is to investigate the usability and usefulness of our impact analysis design and implementation by asking the participant to run some change management scenarios and gathering feedback through answering the questionnaire. During the study, we provided the user with a demonstration of the impact analysis tool and functionality and then we asked the participant to apply for some the tasks. The participants of the study implemented the change scenario as an example in a tool then explored the impact of modification in existing ERP systems. Based on the result of their action we asked the participant to assign and plan change implementation by assuming that they have sufficient knowledge in this field to propose implementation strategies. After proposing the two sets of implantation strategies for each impact item we ask them to run the propose

solution to impact analysis tool so that they can observe the difference in terms of cost and select the most efficient solution to be considered as an implementation plan.

Same as the last session in the previous study with expert we asked the participant to provide feedback about the technology acceptance and usability, i.e., system usefulness, ease of use, ease of learning and satisfaction with the system (Lund 2001). The study involves asking the participant to apply a few exercises by using a change impact analysis tool to evaluate the following usability criteria:

- Learnability: How easy is it for the ERP user to accomplish basic tasks the first time they encounter the impact analysis tool?
- Satisfaction: How pleasant is it to use the impact analysis tool
- Ease of use: Once the ERP user has learned the impact analysis, how many mistakes do ERP users make during preforming tasks, and how easily can they recover from them.
- Usefulness: How useful impact analysis tool for a business analyst in order to understand the impact problem? How quickly and easy can the business analyst performs the impact of change tasks?

The items of the questionnaire were perceived using a Likert scale to gather additional qualitative data from the participants. All items are evaluated on a 5 point Likert scale from strongly disagree (ISO-9126) to strongly agree (5) (see Appendix C).

8.4.3 Data analysis

The study constitutes an important part of our final evaluation of this research. Two data collection methods we conducted in our evolution that is the questionnaire and a short interview. The analysis process is described in two parts. One part involves a qualitative analysis where the focus is on the experience of ERP specialist on the current impact analysis approaches during the change process through the result of the interview. The other part of the study consists of a quantitative analysis of estimating our evaluation criteria (i.e. functionality and usability) through the use of change impact analysis tool.

8.5 Results

8.5.1 ERP Experts Interview Assessment (Part I)

Interviews were transcribed and summarized in the plain text. The transcripts were then read through and analysed in detail (see Appendix E). Analysis of the transcripts focused on the identification of the methodologies, approaches, and techniques and tools that ERP experts employed during the change process. The discussion starts by going through each question separately and analyses the issues including quotes from the interview data to illustrate our findings.

Q1: Any experience using impact analysis tool for ERP system? (e.g. regression testing, SAP solution Manager 7.0, PanayaIA, etc.)

All ERP expert emphasises that there is no particular impact analysis tool that they have used so far to define the impact of modification in ERP systems. Only one of the participants has experienced using SAP solution manager where the purpose is for documentation repository rather than impact analysis, and he quoted that:

> *"We use Solution Manager but not for assessing the impact. We only use it for to documentation repository to process our documentation and not using for process mapping and linking the processes to define the dependencies such as roles and functionality."*

The result of interviews confirms that there is no particular software tool available for the ERP expert to analyse the impact of modification in order to define what ERP components (i.e. business process, function or data) in the ERP system will be affected. This explains that our tool could be a solution in future for the ERP expert to cover some of their requirements during the change management procedure. So to justify this statement we need to explore from the ERP analyst which procedure they are using during the ERP modification and how they were evaluating the impacts in the entire system.

Q2: How do you analyse change propagation in your ERP system? Do you use any informal way (e.g. expert judgment, previous historical data, and previous experience, etc.) or any other formal method or technique?

The purpose of raising this question is to focus on approaches that ERP specialists are embedded during the change that leads the modification of the ERP system.

As most of the ERP experts outlined the strategy in determining change and their impact is more based on expert judgment rather than using any formal mechanisms and tool. During the change process, ERP experts identify the requirements and mapping both the as-is and the to-be process model into process Modelling tools like Microsoft Visio, Bizagi, RPIW (rapid process improvement workshop). Then by using these tools and techniques they can define all the interaction and dependencies in a high-level form to analyse the problem, as expert 4 states that in his interview:

> *"We deal with these situations is to use process mapping tool like Visio or RPIW and gather various employees from a different sector to analyse the problem. "*

However, apart from using Visio as tool all approaches mentioned by the expert always require consultants to investigate the problem and define the impact based on their knowledge, as the expert 6 state that:

> *"We use Visio for documentation, but more everything is based on our judgment during the change process rather than using any particular tool for that."*

This confirms the limitation in the change management process of not having a tool to determine the impact of a change in ERP systems as an automated and productive manner.

Q3: Do you have a standard set of steps/activities or a process to trace the impact of modification in your ERP system?

The result of this question indicates that some experts do not follow any formal approach to assessing the change in ERP system while others have a standard procedure in which they require documenting the whole process during the modification process. According to them first, they translate the requirements by process mapping then asking the business analyst to locate the problems in the process model and define the dependencies and interactions. After that, they evaluate the impact of the new requirement, and determining if the change in the requirement may fit into the ERP system or not. During each task expert indicates that they use formal documentation in order to assess the problem like expert 1 outlines this as follows:

"We assess the impact by implying into the templates that include three-page documentation which needs to go around to the various teams to evaluate the modification deeply including the implementation, cost and training."

Furthermore, one the experts mentioned about applying a technique such as six-sigma methodologies that used as a data-driven approach for eliminating defects in any processes. He stated in his interview about this method:

"We implied the six-sigma methodology to define and locate where the problems are in the system, and then gathering people such as developer analyst to assess if they can fit this modification with the system for this purpose or not."

In all cases, they need a team of specialists to analyse the problem further, however, often to define the impact they spend days and hours in order to clarify the problem and find the alternative solution, as this also mentioned by Expert 3:

"During the change process, 12 people sat down and discussed the problem for about five-day and then mapping the whole thing, indicating where the problems are and considered the solution to apply for additional requirements."

While others hiring this team of consultant from change management organisations to find it quicker and efficient since they do not have the enough knowledge to resolve this issue, as expert 4 outlines this:

"we outsourced change management consultant from another company as a part of our team. Once significant changes identified change management team, have to ensure that everyone can understand the change and what the new system is going to do."

Overall from a discussion with the experts it is useful to follow a constructive way during a modification process, however, this needs to be done with the judgment of the business analyst to locate and assess the problem in detail. Although having a team of consultants during the process is an advantage, this may cost a fortune and time for an organisation to analyse the problem accurately and correctly. Having a tool with a structured change process that the organisation can run during the change process without asking much from an external consultant can save a lot of time and cost to analyse the problem efficiently.

Q4: What kind of information is available in relation to impact analysis and what information do you use during the analysis process

There are two types of documentation mentioned by expert during the change process:

- The first is process mapping documentation that mostly is designed via Microsoft Visio to describe the ERP components dependencies of as-is and to-be business processes.
- The second is the change control documentation in which they stored a change data through the template form with a detailed description of the modification.

As stated by the expert 5 they do not have any dependency model, and everything is more based on analyst experience and judgment. However, as expert 3 discussed further by reusing the process documentation similar as dependency document for future changes. He quoted:

> *"It is important for us to document the whole procedures as-is and to-be of process mapping to reuse it during the modification process."*

Q5: How important for you to use a structured impact analysis approach when assessing the impact of modification in your ERP system?

Almost all experts outline that depending on the solution they believe the impact analysis tool can be an effective and efficient solution for assessing the side effect of the change in the ERP system especially for those organisations with a thousand documentation and dependencies. Besides that, according to the Expert 4 having tool and structured approach may reduce the personal judgment during the change process, during the discussion he quoted:

> *"That would be very useful which help to take the personal experience out of it, and the impact of modification would be assessed in more mechanical, logical and consistent approach."*

Also, the structured approach can eliminate the incorrect analysis when the business analyst is missing some parts to be captured during the assessment. Therefore, having a tool could enhance the measurement more accurately and correctly.

Table 8-5 Summarized result from all ERP expert

	Current position and experience (industry)	Years of experience with ERP (ERP product)	Current tools/methods used in post-implementation change management
Expert 1	Head of ERP Training (Large Pharma)	10 years (Netweaver)	Expert judgment Ad-hoc process version tracking Change control document
Expert 2	Operational Excellence Lead EMEA (Large bank)	16 years (Netweaver)	Expert judgment Ad-hoc process version tacking Change control document
Expert 3	Finance Director (Medium software house)	18 years (SAP, JD Edwards/Oracle)	Expert judgment BPMN process models version tacking Change control document
Expert 4	Senior business analyst (Higher Education)	25 years (SAP, Oracle)	Expert judgment Ad-hoc process version tacking
Expert 5	Management Consultant (Large Consulting)	20 years (SAP, MS Dynamics, JD Edwards/Oracle)	Expert judgment BPMN process models version tacking Change control document
Expert 6	Analyst Developer (Large Consulting)	7 years (Netweaver)	Expert judgment Ad-hoc process version tacking Change control document
Expert 7	Business Analyst (Smets-Solanes and De Carvalho)	5 years (Custom ERP System)	Expert judgment

8.5.2 Result for functionality assessment with ERP expert (Part II-A)

To summarised the above findings, we created a generic model that currently is used during the ERP change process (see Table 8-5). All members of the panel have acknowledged the current lack of structured approaches to ERP Post-implementation change management in their respective organisations. According to our panel, change request management involves a subjective judgment based on written documentation physically circulated among stakeholders or provided in workshops with ERP consultants. This subjective judgment is only qualitative as there is no indication of the quantitative impact of the change nor an established change management process through which requirements are tracked. As far as requirements evolution tracking is concerned, some of the experts reported

the use of ad-hoc version tracking of process models, e.g., MS Visio files. These models, however, have no formal association with the actual ERP installation and they often tend to represent an ideal case for process execution, rather than the processes implemented in the ERP system. In some cases, version tracking considers BPMN process models.

In this section, the results will be analysed and discussed. These evaluations assess the applicability and validity of our research approach. During the data analysis, measuring average response to the questions can provide useful information, which gives an indication of where the majority of the responses are centred. In the following section, we analysed the data from the questionnaire and discussed the results. The table in the following sections uses the same labelling for measurement items as in the questionnaire (see Appendix C Part A). The tables present the distribution of responses categories for each evolution criteria to assess the impact analysis tool.

Frist we asked the ERP expert to answer to the following question:

Q: "Do you believe that impact analysis is missing some items to capture as change impact? If yes, please indicate what? "

Figure 8-10 present the results concerning the missing items indicating that tool could not capture during the analysis. According to the results, more than half of the experts noted that the impact analysis tool should involve other aspects like organisational elements during the change process. We asked the participant to specify the items that they expected that impact analysis should capture during.

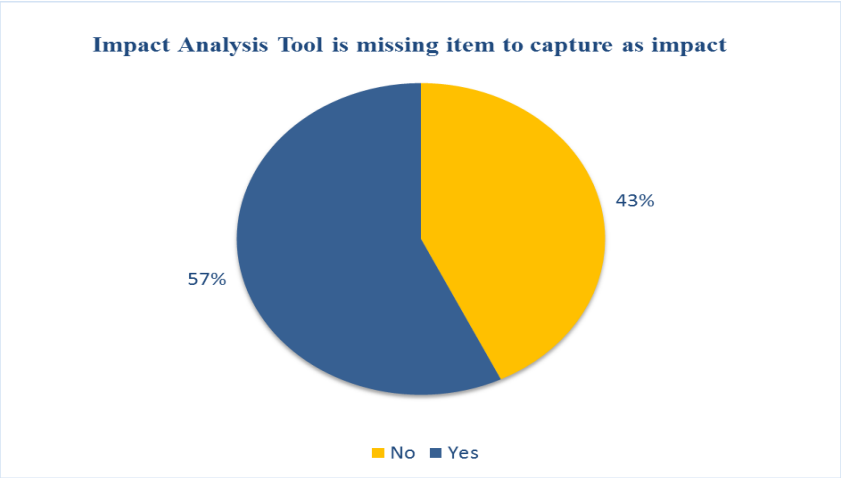


Figure 8-10 Result for missing item

The item that impact analysis is missed according to the ERP expert opinions are listed in following categories:

- GUI impact: The Experts were expected that impact analysis tool could cover the screen shot or specific fields of the forms
- Organisational impact: The expert preferred that impact analysis result demonstrates the organisational area such as plants, division (finance, sales, etc.) in addition to the roles and responsibility of the person who performs the task in the ERP system.
- Authorization impact: Another aspect that experts were noted are the authorization level and security during the modification of the ERP system.
- Financial impact: Expert indicates that the impact analysis tool should provide some financial aspect in terms of cost and expensive for the organisation as a result of the modification. Such as training cost, maintenance cost and effort of implementation cost.

Note that the above listed items mentioned by the ERP specialist are not included within the scope of this research. The tool can be extended in future to include these aspects as a part of analysis. In addition, none of the specialists addressed the issues related to the capturing of existing impact item accuracy.

Another concern during the evolution of impact analysis was to score the accuracy of our impact analysis from low to high shown in table 8-6 and Figure 8-11. We asked:

Q: "Please indicate to what extent you rate the accuracy of impact analysis tool for defining the components?"

The average score for this statement was 4.14 that indicate that the tool provides an accurate result.

Table 8-6 Frequency of Rating Accuracy of Impact Analysis

		Response: (1) Very Low - (5) Very Highy						
Item	List of Criteria	Mean	(1)	(2)	(3)	(4)	(5)	Total % Agree
1	Accuracy of impact analysis	4.14	0.00%	0.00%	14.29%	57.14%	28.57%	85.71%

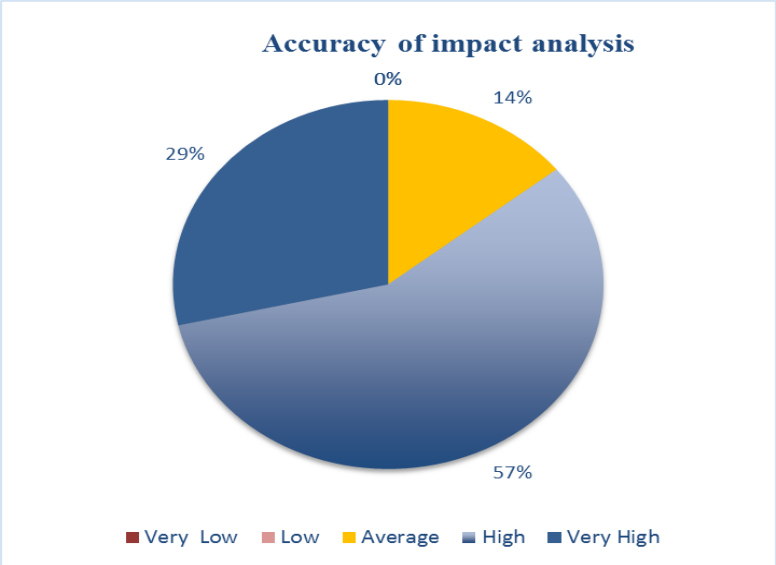


Figure 8-11 Result for rating the accuracy of impact analysis

Overall, from 6 out of 7 experts we perceived that the tool produces highly accurate result in order to analyse the effect of the change in the ERP system. Based on the results, we believe that the tool offers the functionality for business analyst in order to assess the modification effectively.

8.5.2.1 Functionality criteria

Table 8-7 presents the results of the measurement for functionality criteria. The result sorted based on the calculation of the mean from low to high (see Figure 8-12). The last column of the table below indicates that the percentage of the respondent that agreed with the

statement about the functionality tool and accurate result. From the result, we perceived no disagreement by any of the expert on any of the statement. In addition, all the participant agreed that this method could improve the ERP customization more effectively and enhance the decision making for planning the change.

Table 8-7 Frequency of Rating Functionality of Impact Analysis

Item	List of Criteria	Mean	Response: (1) Strongly Disagree - (5) Strongly Agree (1)	(2)	(3)	(4)	(5)	Total % Agree
C1	The impact estimate is a suitable measurement for implementation	3.86	0.00%	0.00%	28.57%	57.14%	14.29%	71.43%
C2	The cost estimate is a suitable measurement for comparing implementation strategies	3.86	0.00%	0.00%	28.57%	57.14%	14.29%	71.43%
C3	The method improves the ERP customization more effectively	4.00	0.00%	0.00%	0.00%	100.00%	0.00%	100.00%
C4	The method enhances decision making for planning the change implementation	4.14	0.00%	0.00%	0.00%	85.71%	14.29%	100.00%
C5	The tool identifies all affected ERP components	4.14	0.00%	0.00%	0.00%	85.71%	14.29%	100.00%
C6	The tool identifies impact set correctly	4.14	0.00%	0.00%	0.00%	85.71%	14.29%	100.00%
C7	The tool identifies all occurring impact items	4.14	0.00%	0.00%	0.00%	85.71%	14.29%	100.00%
C8	The tool analyses the impact effectively	4.14	0.00%	0.00%	14.29%	57.14%	28.57%	85.71%
C9	The tool classifies impact item into the ERP components categories	4.29	0.00%	0.00%	0.00%	71.43%	28.57%	100.00%
	Total	4.08	0.00%	0.00%	7.94%	76.19%	15.88%	92.06%

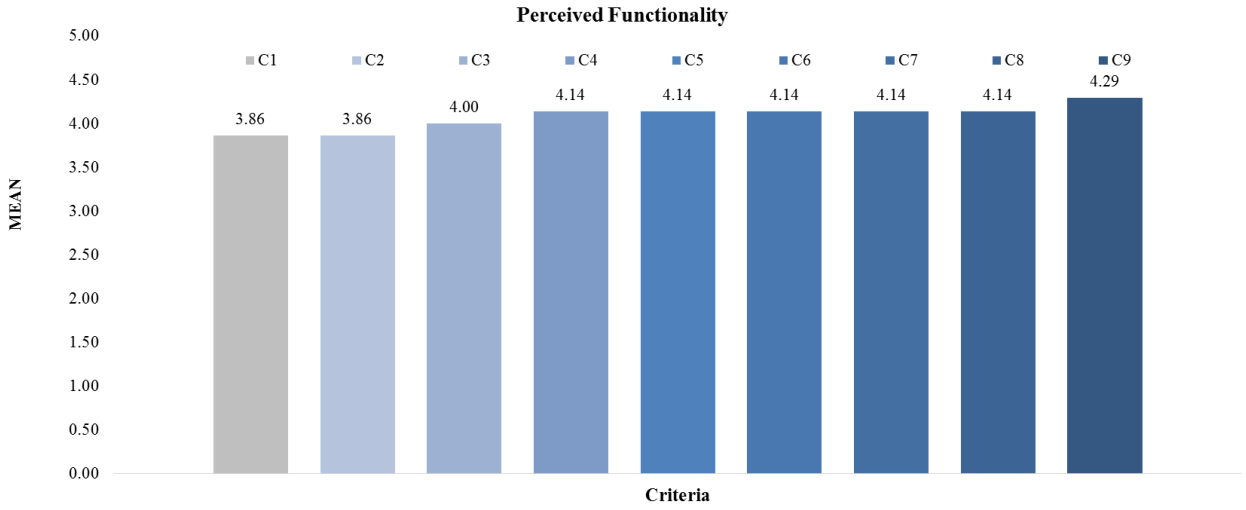


Figure 8-12 Result of impact analysis functionality

From the above table C1, and C2, which stresses on the measurement of the estimate for impact assessment and cost assessment we have received two of the respondents answered neutrally to this statement. They both stated that the measurement should also include the maintenance effort such as the cost of training the ERP end user and the cost of gathering resources for implementation like hours of hiring a developer or consultant.

Moreover, they indicate that impact should include the assessment concerning the percentage of the department. This implies that to what degree different department will be impacted.

8.5.3 Result for Usability assessment with both group (Part II-B)

To evaluate the usability of the tool outlined in this thesis, a questionnaire was used and sent to the different user groups of the tool. In total, 19 samples were collected. Seven of the respondents were ERP experts using the tool to specify analysis frameworks, and 10 were former students who used the ERP tool as part of their education. This section presents the result of both studies with the ERP expert and non-expert on the usability aspect of our impact analysis tool.

8.5.3.1 Usefulness

The study also asked the participant from both groups to answer the questions about the usefulness of impact analysis tool. Figure 8-13 shows a summary of the usefulness criteria through the calculation of means for each group category and Table 8-8 shows the frequency of the responses. The results of both groups indicate that impact analysis can be useful for defining the modification impact. The result also shows that the participant can understand and identify the impact easily and effectively. However, 4 out of 7 among the experts indicate that they expected more functionality from impact analysis tool and all highlight the points where they referred as missing items during the assessment of impact analysis functionality. Such that they were expected that impact analysis could provide them with information about the organisational data modification like role and responsibility or division in the organisation in addition to that so they can have more clear view of the impact from the various aspect in the enterprise model. Upon that the result forms the test indicate that our impact analysis tool can also be useful for a non-ERP expert who has limited knowledge of ERP system to run the impact analysis tool. The can easily understand the consequence of the action during the ERP modification, and they can easily identify the impact through the impact analysis tool.

Table 8-8 Perceived Usefulness

Item	List of Criteria (Usefulness)	Group	Mean	Response: (1) Strongly Disagree - (5) Strongly Agree (1)	(2)	(3)	(4)	(5)
C1	Does everything that I would expected	NE	4.08			25% (3)	42% (5)	33% (4)
		EE	3.29		29% (2)	29% (2)	29% (2)	14% (1)
C2	Helps me to be more productive	NE	4.25				75% (9)	25% (3)
		EE	4.14			14% (1)	57% (4)	29% (2)
C3	Understand the effect of my action during ERP Modification	NE	4.25				75% (9)	25% (3)
		EE	4.29				71% (5)	29% (2)
C4	Helps me understand the impact easily	NE	4.00			8% (1)	84% (10)	8% (1)
		EE	4.29				71% (5)	29% (2)
C5	Meet the requirements to identify the propagation of change	NE	4.33				67% (8)	33% (4)
		EE	4.14				86% (6)	14% (1)
C6	Helps me to identify impacts more effectively	NE	4.25				75% (9)	25% (3)
		EE	4.29				71% (5)	29% (2)
C7	Gives me more control over the activity when using ERP system	NE	4.25			8% (1)	58% (7)	34% (4)
		EE	4.29				71% (5)	29% (2)
C8	Saves me time of identifying impact	NE	4.42			8% (1)	42% (5)	50% (6)
		EE	4.43				57% (4)	43% (3)
	NE =Non Expert, EE= ERP Expert			Strongly Disagr	Disagree	Neutral	Agree	Strongly Agree

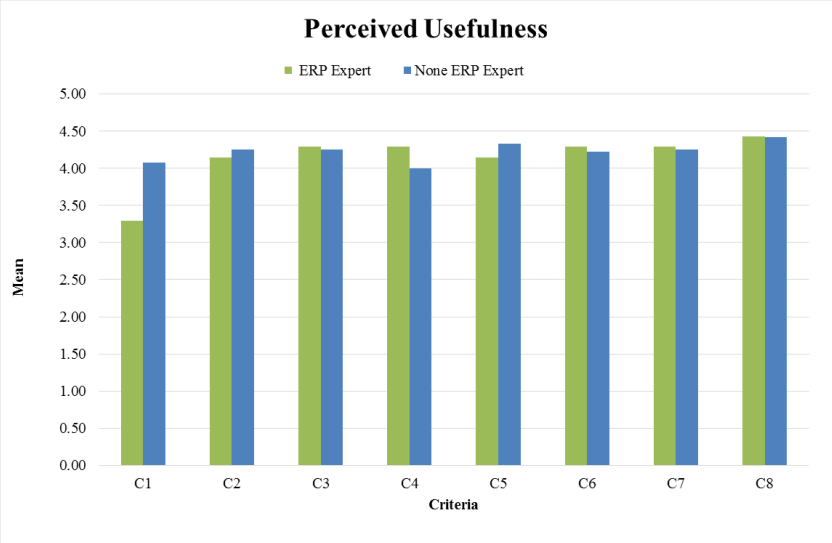


Figure 8-13 Perceived Average Usefulness

8.5.3.2 Ease of Use

The frequencies of responses for the criteria to perceive the ease of use of impact analysis are presented in table 8-9. The criteria 1 indicate the complexity of impact analysis tool. From the ERP expert point of view 4 out of 7 agreed that impact analysis is unnecessary complex and only one of the experts has some difficulties in order to use the tool whereas the others especially students found the impact analysis tool as complex. This is because the

students have very limited understanding of ERP system functionalities and business processes compare to the ERP expert therefore they found it complicated.

Table 8-9 Perceived Ease of Use

Item	List of Criteria (Ease of use)	Group	Mean	Response: (1) Strongly Disagree - (5) Strongly Agree (1)	(2)	(3)	(4)	(5)
C1	Impact analysis is unnecessary complex	NE	3.00	17% (2)	25% (3)	8% (1)	42% (5)	8% (1)
		EE	2.43	14% (1)	43% (3)	29% (2)	14% (1)	
C2	Fewest steps is required to accomplish the task	NE	3.83			36% (4)	55% (6)	17% (2)
		EE	3.57		14% (1)	29% (2)	43% (3)	14% (1)
C3	Recover from mistake quickly	NE	3.92			17% (2)	75% (9)	8% (1)
		EE	4.00			29% (2)	43% (3)	29% (2)
C4	Confidently use the tool	NE	3.83			17% (2)	83% (10)	
		EE	4.29			14% (1)	43% (3)	43% (3)
C5	Tool is user friendly	NE	3.75		17% (2)	17% (2)	42% (5)	25% (3)
		EE	3.86			43% (3)	29% (2)	29% (2)
C6	Procedure of running impact analysis is easy to follow	NE	4.00			8% (1)	84% (10)	8% (1)
		EE	4.14			14% (1)	57% (4)	29% (2)
C7	Features are well integrated	NE	4.00			17% (2)	67% (8)	17% (2)
		EE	4.14			14% (1)	57% (4)	29% (2)
	NE =Non Expert, EE= ERP Expert			Strongly Disagr	Disagree	Neutral	Agree	Strongly Agree

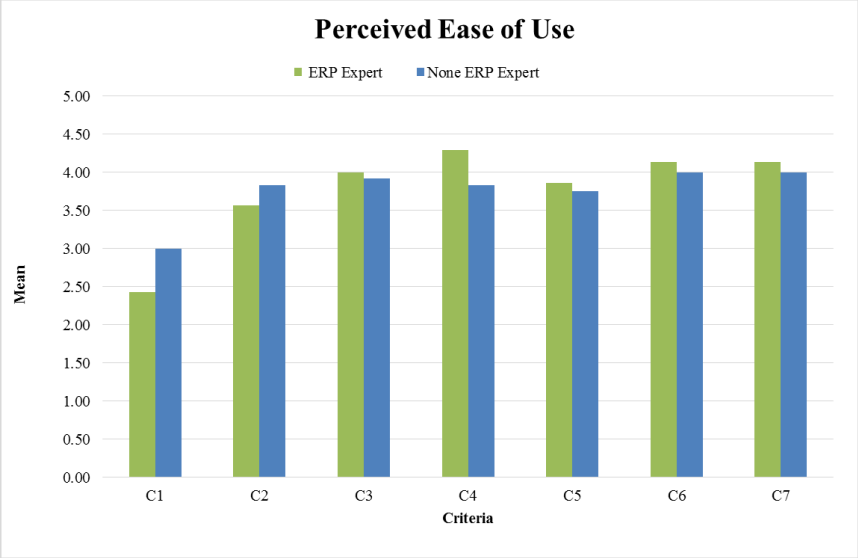


Figure 8-14 Perceived average ease of use

In addition, the tool has not been designed in such way to be user friendly therefore some of the participants might found it difficult to perform their task. This also shows in our result that only 8 out of 12 students and 4 out of 7 expert agreed that impact analysis tool is user friendly. In addition to the complexity point some of the participants indicate that they had to go through more than few steps in order to accomplish the impact analysis task.

However, majority of the participants from both groups agreed that the procedure of running impact analysis is easy to follow and the features of impact analysis tool are well integrated.

8.5.3.3 Ease of learning

The responses to the criteria to perceive the ease of learning of impact analysis are presented in table 8-10. The result indicates that the procedure in order to user the tool is easy and the user from both users did learn how to use the tool quickly. We perceived no disagreement result from any groups. However, the result show that the student responded to these criteria more strongly than ERP expert. This is due reason that the expert expected more facilitation like help functionality as a feature in order assist them during the impact analysis task.

Table 8-10 Perceived Ease of learn

Item	List of Criteria (Ease of Learn)	Group	Mean	Response: (1) Strongly Disagree - (5) Strongly Agree (1)	(2)	(3)	(4)	(5)
C1	Easily remember how to use tool	NE	4.33			8% (1)	50% (6)	42% (5)
		EE	4.00			14% (1)	71% (5)	14% (1)
C2	Learned to used quickly	NE	4.50			8% (1)	33% (4)	58% (7)
		EE	4.14			14% (1)	57% (4)	29% (2)
C3	Tool is easy to learn	NE	4.33			8% (1)	50% (6)	42% (5)
		EE	4.14			14% (1)	57% (4)	29% (2)
NE =Non Expert, EE= ERP Expert				Strongly Disagr	Disagree	Neutral	Agree	Strongly Agree

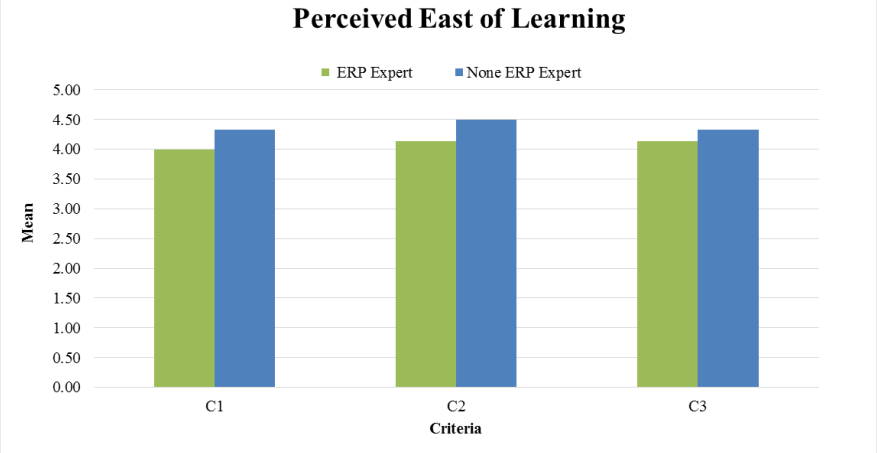


Figure 8-15 Perceived average ease of learn

8.5.3.4 Satisfaction

In table 8-11 participants from both groups positively agreed that the impact analysis tool can improve the quality of work for ERP practitioner to identify the effect of change and understand the dependencies in the ERP system. Also from the experiment we perceived almost all participants were satisfied with the tool and suggested this to the ERP expert to use in the future for ERP customization.

Table 8-11 Perceived satisfaction

Item	List of Criteria (Satisfaction)	Gourp	Mean	Response: (1) Strongly Disagree - (5) Strongly Agree (1)	(2)	(3)	(4)	(5)
C1	Works the way that I expected	NE	4.33			8% (1)	50% (6)	42% (5)
		EE	3.43		14% (1)	29% (2)	57% (4)	
C2	Recommend it to ERP expert	NE	4.42			8% (1)	42% (5)	50% (6)
		EE	3.71			29% (2)	71% (5)	
C3	Can improves the quality of work	NE	4.42				58% (7)	42% (5)
		EE	4.00				100% (7)	
C4	Satisfied with tool	NE	4.50				50% (6)	50% (6)
		EE	3.86			14% (1)	86% (6)	
	NE =Non Expert, EE= ERP Expert			Strongly Disagr	Disagree	Neutral	Agree	Strongly Agree

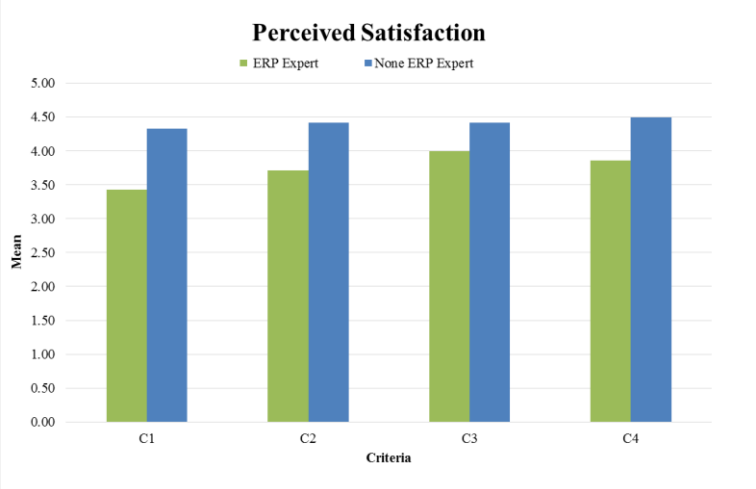


Figure 8-16 Perceived average satisfaction

Although some of the ERP experts had more expectation from the impact analysis tool compares to the non-expert ERP. This is because the ERP experts have a broad view of ERP system customization than other participants, which need them to seek for more

information from the tool. Some of these experts outline this information can be organisational elements (i.e. role, organisational areas, division), GUI (i.e. wizard), and the technical assessment in order to complete the analysis of the change in the ERP system.

8.5.3.5 Discussion

Overall the results confirm that the users strongly agree or agree on the fitness of the solution to change management criteria of ERP system post-implementation. In both cases, the respondents agree on the occurrence of the impacts. The result from the questionnaire explains that the change impact analysis approach identifies the impacts and the affected entities. This helps the user to understand the consequences of the modification they request before they implement them permanently in the system.

The feedback obtained from both groups of participants is overall positive. The few unsatisfactory results are found in the evaluation of ease of use. Because of its proof of concept nature, our tool was not developed considering usability as primary non-functional requirements. Therefore, lower scores for the ease of use criteria have been expected.

In general, the responses from both groups are encouraging. The participants agree that the alternative strategy selection is helpful to understand the cost implication when a change is going to be implemented and is useful to select the best strategy through proposing various implementation solutions. Some of these users; however, focused on the presentation of the impact analysis (user interface issue) which is not the primary concern of the evaluation.

The participants further give the following feedback on the usability of the impact analysis tool:

- Providing a better interface to enable the users to assign all the alternative strategies at a different level of modification in parallel in a form of a single view rather than for each individually.
- The impact analysis needs to be flexible and customizable in order to assign the different weight to calculate the severity of the impacts of the system.

The student group has provided generally higher evaluations. This can be explained by their relative lack of experience with utilizing ERP systems in the real world, which has

prevented them to realize the degrees of the issues that can be encountered in ERP post-implementation change management in real world scenarios. Whereas, the expert participants stressed that the tool could be extended with more advanced functionality to assess the organisational impact of ERP changes and the financial impact of the proposed changes based on different possible implementation strategies.

8.6 Summary

This chapter provides empirical evidence to validate our approach and impact analysis tool. It explains the evaluations goal, evaluation criteria, and the evaluation method in order to meet our objective. First two ERP case studies demonstrated from two different systems then presented the result to that addressed the feasibility and applicability of applying impact analysis tool. However, some of the specific features (i.e. Journal for Microsoft dynamic NAV) could not be captured by our impact analysis since this requires further configuration and adjustment to map all the components for these particular ERP systems.

Then this chapter evaluated the applicability and functionality of our approach and tool through the study with ERP expert. The interview with ERP expert indicates that there is no tool available with the functionality to assess the impact of modification that the practitioners can be used during the time of ERP customization and mostly the assessment of modification is done through the expert judgment. Further the result of our experiment with ERP expert for assessing the functionality of our tool reveals that the ERP expert mostly agreed that the tool can improve the ERP customization and enhances the decision making for planning the change more effectively.

Finally, this chapter presented the assessment result of the usability experiment of our tool by ERP experts and non-experts (i.e. student). The result has shown that our impact analysis tool reaches a degree of usability in which both non-expert and expert found the tool useful and easy to use.

CHAPTER 9 CONCLUSIONS

9.1 Introduction

This chapter draws the conclusions of our research work. It starts with a summary and explanation of how the research achieves each objective set earlier. This is followed by a discussion of the main contributions of this research in response to the changing process in ERP system evolution. Then addressing some limitations in the current scope and introducing the areas of future work conclude this chapter.

9.2 Research Summary and Achievement

This thesis presents a framework for change impact that can be applied in the context of enterprise system post-implementation. The presented approach supports business analyst and developers performing post-implementation and modification task that involve frequent changes of existing and potentially long-living ERP system. It provides an estimate of the impacts of a modification and provides an estimate cost for planning the implementation of the change and decided between alternative solutions based on their impact before the change is actually being implemented.

By using the approach and techniques developed in this thesis stakeholders can potentially identify the impacts before implementing a change in the ERP system, which can significantly reduce the risks of intervening the system and the cost of implementation. As a result, our approach can help business analysts, project managers, system developers and maintainers plan changes, make changes more accurately, accommodate certain types of system changes, and trace through the effects of changes. They can also use this approach to evaluate the appropriateness of proposed modification. If a proposed change has the possibility of impacting a significant part of the system, then they can think of other alternative solution to determine a safer change into the system.

The overall framework accommodates how to handle the change and provides better assessment approach for analysing new requirement in the ERP system.

- The project managers can employ this impact analysis tool to run "what if" analyses on different modification proposals to assess the risk, and potentially select the appropriate implementation strategies that is most cost effective.
- The system developers can use this technique to indicate the vulnerability of critical components of the ERP system. For instance, if a business function that provides critical functionality in the system is dependent on many different parts of the system, therefore, this component is susceptible to modifications.
- The system testers can apply this technique to define which areas are impacted by the modifications and allowing them to focus only on those components and feel confident about the quality of operating the system after implementation of change.
- The business analyst can use this method to evaluate different ways of accommodating end user requirements in the system by proposing several solutions and assessing the impact.

A set of research objectives as defined in Chapter 1 represents the direction of this thesis. Hence, these objectives are highlighted again to observe and summarize how each has been implemented and achieved.

1. *Develop a generic conceptual meta-model of ERP systems to determine the dependencies among the different components constituting the system;*

Identifying dependencies in ERP systems is essential to ensure adequate change management process. As described in Chapter 4, Dependency relationships between ERP system components are captured based on the ERP dependency meta-model that contains all set of components at the enterprise level such as business processes, business functions, etc. The dependency meta-model of ERP system is designed in the form of a UML class diagram in which usually each ERP component denotes as an entity and each association lines as a dependency relationship between entities. Then we used this meta-model by two different ERP system case study to prove the applicability of using this model to any ERP system.

2. *Introduce taxonomy of possible post-implementation modifications of ERP systems;*

Based on the dependencies defined by the conceptual meta-model the taxonomy of modification is presented which allows to understand the objectives, and to determine the particular needs of developing a proposed change in the ERP system. We classify ERP modifications along three dimensions, i.e. the level, type of change and the granularity of change operation. Chapter 4 is shown how change operations can be modelled and classified in order to provide automated change impact analysis run.

3. *Define a methodology to assess the impact of different types of change, by considering, in particular, the ripple effects implied by specific dependencies;*

Impact analysis determines the scope of the modification and the complexity of the modification. Based on the change taxonomy defined in Chapter 4, we describe an impact analysis algorithm for each type of change, i.e., add/delete/update of data object/function/process. We classified the impact items into two stages of design time, and run-time, and we specify an action in order to manage the modification for ongoing transactions at the run-time stage. Chapter 5 describes the impact analysis mechanisms, which capture the ripple effects of ERP modifications on the existing design-time and run-time structure of the ERP system.

4. *Define metrics to estimate the depth of the impact of ERP post-implementation change, possibly based on the strategy selected to implement the identified change.*

As Chapter 5 discussed we present set of metrics for assessing the impact of a proposed ERP change. These metrics aim at enhancing the decision making to plan the implementation of the modification efficiently. The metric estimate propagation to quantify

the impact regarding the entities modified at design/run time and, based on that, define a risk level for the proposed change. Also, the metrics can predict the effort for change implementation based on the cost of proposed strategies to implement the modification.

5. *Implement a software tool, i.e., a decision support system, embodying the identified models, methods, and metrics to support business analysts in the controlled management of ERP post-implementation change.*

We present the design and implementation of the impact analysis tool as proof of concept to demonstrate the feasibility of our impact analysis methodology. Chapter 7 described some of the feature developed to support change impact analysis. The tool is designed through a model-driven approach that aimed to automate the useful information for change impact analysis, which includes the mechanisms for assessing the impact, the propagation metrics and cost implication of modification.

9.3 Contribution

A methodology presented in this thesis provides business analysts and practitioners with scientifically grounded method to manage ERP post implementation modification in controlled manner. The application of our approach improved change impact analysis; reduce the risk associated to post-implementation change management.

This project has important implications for both academic and practice.

As far as academic research is concerned, the results of this thesis contribute to the research field of ERP post-implementation. While this issue has been tackled in the literature mainly at the level of business/management constructs (Ram, Corkindale, and Wu 2013, Oseni et al. 2014), this thesis tackled the same issue for the first time from the design/industrial engineering standpoint. The purpose of this research work is, in fact, to devise concrete methods and tools to support the relevant stakeholders, such as business analysts, strategists and IT designers, in understanding the impact of proposed ERP post-implementation and deciding to what extent the change should be implemented. In other words, while previous research has mainly focused on explaining ex-post the impact of poor

post-implementation change management in ERP implementation, this thesis focuses on the engineering of solutions to improve current practice in ERP post-implementation change management. This latter perspective on ERP post-implementation is new in the literature and has the potential to establish a whole new area of research.

- Our approach supports the precise Modelling of change operation that acts as trigger for actual change impact analysis. We extend change Modelling by introducing and enhanced concept that is based on atomic and composite operations. The proposed atomic change constitutes the basic unit of change while composite operation is comprised other atomic and compost unit of change. The precise Modelling of change operations also allows us to model the exact type of impacts that are determined by our impact analysis mechanisms.
- Our impact analysis approach is based on the analysis of dependency relations connecting ERP components. These dependencies first of all have to be elicited and explicitly mapped. Yet current research does not provide an automated way for mapping dependency from the ERP system. However, it defines a mechanism through a generic dependency meta-model of ERP components in order to, determine the type of dependency during mapping process.
- The presented impact analysis approach is based on how the effects of change propagates across dependencies to related ERP components. The proposed approach analysed this interaction using a set of impact mechanism that are designed to react on certain change operations and dependencies. The impact mechanisms are triggered by change and are able to determine how these change affect related ERP components. Our approach is able to predict the propagation of change across different level of modification.
- Our tool implements and supports the phases of our approach and currently allows for change impact analysis of ERP system. With the help of our tool we conducted an evaluation of our approach, and the comprehensive evaluation reposted in this thesis. We assess our approach against the existing approaches and techniques used for analysing the impact of change at ERP system. We then assess the functionality and

applicability of our tool and approach with ERP expert in order to enhance the performance and productivity of activity of the change during maintenance task.

This thesis provides a framework for ERP post-implementation change management which has the capability to employ on any ERP systems. The proposed framework shares some commonalities with other methodologies in change management. However, their methods give a design-time snapshot and typically does not involve any run-time concerns. On the contrary, our framework suggests strategies on the process instances affected by the change to terminate them safely and to plan the change implementation. As part of the proof of concept, the artefacts defined by our framework have been utilized in a software tool (i.e., a decision support system) supporting impact analysis and assessment. The implemented tool has been instantiated in the case of two commercial ERP systems and evaluated in practice by ERP experts that reveal our framework provides an effective solution. The proposed framework is the first of its kind supporting a constructive approach solving the practical issues of post-implementation change management for complex enterprise systems. Adopting this method by other researcher and developer with a similar interest in change management can enhance them to leverage this framework through supporting their design approach or assess how well they developed their tool.

This project has also the potential to make huge impact for practice. Most medium and large organisations have already gone through at least one ERP implementation cycle and find themselves in the post-implementation phase. Business analysts in the IT function of medium/large organisation or in consulting companies struggle daily to address the need to manage ERP post-implementation change. This project will provide business analysts with a scientifically grounded method and software tools to support the management of ERP post-implementation change in a controlled way.

In the longer term, the results of this project can also be extended to other classes of enterprise systems, such as Customer Relationship Management (CRM) and Product Lifecycle Management (PLM), and to the context of change impact analysis in Enterprise Architecture (EA).

9.4 Research Limitation and Future Research

Despite the above contributions, the findings are also exposed to some limitations as follow:

Limitations of Methods

Our design process has been driven entirely by researchers, i.e., the author. The practitioners have been involved only in the evaluation phase; they could have been involved in the design as well (e.g., to validate intermediate requirements). While such involving only researchers in the design phases is legitimate in design science research, the literature also advocates for a more direct involvement of practitioners in the design of artefacts. This limitation is mitigated by the evaluation of the proposed artefacts in real world settings.

Our approach is not entirely explicitly formalized and relies on some assumptions that may not be valid in practice (e.g., a check for compatibility of data, function, processes must be available; the definition critical point is not formally defined for all possible patterns in a process); evaluation is done with pseudo-real companies (Cronus and GBI) and not with actual real companies

Another limitation concerns the model-driven approach underpinning the different phases of the methodology. The use of models essentially entails on the one hand that all relevant aspects of a domain must be captured in a model and, on the other hand, that any aspects not directly captured by a model is excluded from the analysis. In the case of our methodology, this is particularly relevant when capturing dependencies among ERP components. This could potentially be a very time consuming and cumbersome process and any aspect not captured in the dependency meta-model is prevented to be considered in the impact analysis and assessment phases. A possible solution to this issue is to reconstruct the dependency meta-model directly from information available in the reference model or blueprints of ERP systems, through data crawling or process mining technology. In this way, only the actual relations among ERP components are identified as dependencies. Such an approach, however, presents multiple challenges, such as the quality of the data available and of the algorithms to identify dependencies.

According to the taxonomy of change explained in Chapter 3, a request may involve modification of one or many ERP components through the analysis of the requirement to define the atomic change and prioritize them in order. Our impact analysis tool is designed in a way that can only define the effects of atomic change per analysis for a change request. This is because some impact items overlap with each other for two atomic changes during the assessment of one change request. Therefore, to address this problem the assumption is made based on the prioritization of the atomic change. The business analysis investigates the assessment of atomic change in order, and if the result of impact for each atomic change is extensively significant, then, the analyst can stop the analysing the remaining atomic change

Limitations of the Results

The validity of the result obtained in this research has been established. Nevertheless, further research could be conducted in this research topic domain. Indeed, confidence in the result could be improved by applying more tests to do real hypothesis test (not only qualitative evaluation).

Future work should focus on developing case studies of real ERP systems. In this way the methodology can gain deeper understanding of the domain. With more time and resources, the methodology should be tested in other complex environment. Another aspect of this research that has to be refined is that using the accurate constant for metrics developed by conducting more experiment with ERP expert.

Since ERP is a large system with further complexity, the usefulness of our approach is tested and applicable to the small size of integration in the ERP system with less complexity. Larger size integration may involve more complex scenarios and other integrated applications of different platforms and environments such as CRM (customer relationship management), SCM (supply chain management) that requires further investigation and analysis.

9.5 Publication

The work presented in this thesis has led to the following academic publications:

M. Parhizkar and M. Comuzzi "*Impact analysis of ERP post-implementation modifications: design, tool support and evaluation*" Computers in Industry 84 (2017): 25-38.

M. Comuzzi and M. Parhizkar "*A methodology for enterprise systems post-implementation change management*" submitted for publication in Industrial Management and Data Systems journal in April 2016.

M. Parhizkar and M. Comuzzi (2016) "*A framework for impact analysis of post-implementation Enterprise Resource Planning modifications*", Proc. SAI Computing Conference 2016 (pp. 706-714). IEEE.

M. Parhizkar and M. Comuzzi (2015) *"An AHP-Based analysis of the cost of ERP Modification"* Proc. 3rd International Conference on Enterprise Systems, (pp. 200 -205.) IEEE

REFERENCES

Abgaz, Yalemisew Mintsnote. 2013. "Change impact analysis for evolving ontology-based content management." Dublin City University.

Abgaza, Yalemisew, Muhammad Javeda, and Claus Pahla. 2013

. "Change Impact Analysis and Optimization in Ontology-based Content Management Systems."

Akkermans, Henk A, Paul Bogerd, Enver Yücesan, and Luk N Van Wassenhove. 2003. "The impact of ERP on supply chain management: Exploratory findings from a European Delphi study." *European Journal of Operational Research* 146 (2):284-301.

Akkermans, Henk, and Kees van Helden. 2002. "Vicious and virtuous cycles in ERP implementation: a case study of interrelations between critical success factors." *European Journal of Information Systems* 11 (1):35-46.

Akkiraju, Rama, and Hendrik van Geel. 2010. "Estimating the cost of developing customizations to packaged application software using service oriented architecture." Web Services (ICWS), 2010 IEEE International Conference on.

Al-Mashari, Majed, Abdullah Al-Mudimigh, and Mohamed Zairi. 2003. "Enterprise resource planning: A taxonomy of critical factors." *European journal of operational research* 146 (2):352-364.

Alawattage, Chandana, Trevor Hopper, Danture Wickramasinghe, Ahmed OR Kholeif, Magdy Abdel-Kader, and Michael Sherer. 2007. "ERP customization failure: Institutionalized accounting practices, power relations and market forces." *Journal of Accounting & Organisational Change* 3 (3):250-269.

Aldin, Laden, and Sergio de Cesare. 2011. "A literature review on business process modelling: new frontiers of reusability." *Enterprise Information Systems* 5 (3):359-383.

Amjad Alam, Khubaib, Rodina Binti Ahmad, and Maria Akhtar. 2014. "Change Impact analysis and propagation in service based business process management systems preliminary results from a systematic review." Software Engineering Conference (MySEC), 2014 8th Malaysian.

Arteta, B. M., and R. E. Giachetti. 2004. "A measure of agility as the complexity of the enterprise system." *Robotics and computer-integrated manufacturing* 20 (6):495-503.

Aryani, Amir, Ian D Peake, Margaret Hamilton, Heinz Schmidt, and Michael Winikoff. 2009. "Change propagation analysis using domain information." Software Engineering Conference, 2009. ASWEC'09. Australian.

Beatty, Robert C, and Craig D Williams. 2006. "ERP II: best practices for successfully implementing an ERP upgrade." *Communications of the ACM* 49 (3):105-109.

Bellahsene, Zohra, Angela Bonifati, and Erhard Rahm. 2011. *Schema matching and mapping*. Vol. 57: Springer.

Bengtsson, PerOlof, Nico Lassing, Jan Bosch, and Hans van Vliet. 2004. "Architecture-level modifiability analysis (ALMA)." *Journal of Systems and Software* 69 (1):129-147.

Bennett, KH. 1990. "The software maintenance of large software systems: management, methods and tools." In *Software Engineering for Large Software Systems*, 1-26. Springer.

Beydeda, Sami, Matthias Book, and Volker Gruhn. 2005. *Model-driven software development*. Vol. 15: Springer.

Bézivin, Jean. 2006. "Model driven engineering: An emerging technical space." In *Generative and transformational techniques in software engineering*, 36-64. Springer.

Bingi, Prasad, Maneesh K. Sharma, and Jayanth K. Godla. 1999. "Critical issues affecting an ERP implementation." *IS Management* 16 (3):7-14.

Biswas, Swarnendu, Rajib Mall, Manoranjan Satpathy, and Srihari Sukumaran. 2011. "Regression test selection techniques: A survey." *Informatica* 35 (3).

Bohner, Shawn A. 1996. "Impact analysis in the software change process: A year 2000 perspective." Software Maintenance 1996, Proceedings., International Conference on.

Bohner, Shawn A. 2002. "Software change impacts-an evolving perspective." Software Maintenance, 2002. Proceedings. International Conference on.

Bohner, Shawn A, and RS Arnold. 1991. "Software change impact analysis for design evolution." Proceedings of the 8th International Conference on Software Maintenance and Re-engineering.

Brehm, Lars, Armin Heinzl, and M Lynne Markus. 2001. "Tailoring ERP systems: a spectrum of choices and their implications." System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on.

Brereton, Pearl, Barbara A Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2007. "Lessons from applying the systematic literature review process within the software engineering domain." *Journal of systems and software* 80 (4):571-583.

Briand, Lionel C. 2007. "A critical analysis of empirical research in software testing." Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on.

Casati, Fabio, Stefano Ceri, Barbara Pernici, and Giuseppe Pozzi. 1998. "Workflow evolution." *Data & Knowledge Engineering* 24 (3):211-238.

Chan, Brian, King Chun Foo, Lionel Marks, and Ying Zou. 2009. "An approach for estimating the time needed to perform code changes in business applications." *International journal on software tools for technology transfer* 11 (6):503-515.

Chen, Injazz J. 2001. "Planning for ERP systems: analysis and future trend." *Business process management journal* 7 (5):374-386.

Choudhury, Islam, Sergio de Cesare, and Emily Di Florido. 2008. "an object-oriented abstraction Mechanism for Generic Enterprise Modelling." *International Journal of Enterprise Information Systems* 4 (1):48.

Comuzzi, Marco, Jochem Vonk, and Paul Grefen. 2012. "Measures and mechanisms for process monitoring in evolving business networks." *Data & knowledge engineering* 71 (1):1-28.

Crnkovic, Ivica, Ulf Asklund, and Annita Persson Dahlqvist. 2003. *Implementing and integrating product data management and software configuration management*: Artech House.

Da Silva, Roberto, Raquel Stasiu, Viviane Moreira Orengo, and Carlos a Heuser. 2007. "Measuring quality of similarity functions in approximate data matching." *Journal of Informetrics* 1 (1):35-46.

Dai, W, D Covvey, P Alencar, and D Cowan. 2009. "Lightweight query-based analysis of workflow process dependencies." *Journal of Systems and Software* 82 (6):915-931.

Daneva, Maya, and Roel Wieringa. 2008. "Cost estimation for cross-organisational ERP projects: research perspectives." *Software Quality Journal* 16 (3):459-481.

Davenport, Thomas H. 1998. "Putting the enterprise into the enterprise system." *Harvard business review* 76 (4).

de Boer, Frank S, Marcello M Bonsangue, LPJ Groenewegen, AW Stam, S Stevens, and L Van der Torre. 2005. "Change impact analysis of enterprise architectures." Information Reuse and Integration, Conf, 2005. IRI-2005 IEEE International Conference on.

De Boer, Luitzen, Jeroen Harink, and Govert Heijboer. 2002. "A conceptual model for assessing the impact of electronic procurement." *European Journal of Purchasing & Supply Management* 8 (1):25-33.

Dijkman, Remco, Marlon Dumas, and Luciano García-Bañuelos. 2009. "Graph matching algorithms for business process model similarity search." In *Business Process Management*, 48-63. Springer.

Dijkman, Remco, Marlon Dumas, Boudewijn Van Dongen, Reina Käärik, and Jan Mendling. 2011. "Similarity of business process models: Metrics and evaluation." *Information Systems* 36 (2):498-516.

Dong, Linying. 2000. "A model for enterprise systems implementation: top management influences on implementation effectiveness." *AMCIS 2000 Proceedings*:13.

Eckert, Claudia, P John Clarkson, and Winfried Zanker. 2004. "Change and customisation in complex engineering domains." *Research in engineering design* 15 (1):1-21.

Ernst, Michael D. 2003. "Static and dynamic analysis: Synergy and duality." WODA 2003: ICSE Workshop on Dynamic Analysis.

Escobar, María Teresa, and José María Moreno-jiménez. 2007. "Aggregation of individual preference structures in AHP-group decision making." *Group Decision and Negotiation* 16 (4):287-301.

Faulkner, Laura. 2003. "Beyond the five-user assumption: Benefits of increased sample sizes in usability testing." *Behaviour Research Methods, Instruments, & Computers* 35 (3):379-383.

Feng, Tie, and Jonathan I Maletic. 2006. "Applying dynamic change impact analysis in component-based architecture design." Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2006. SNPD 2006. Seventh ACIS International Conference on.

Finney, Sherry, and Martin Corbett. 2007. "ERP implementation: a compilation and analysis of critical success factors." *Business Process Management Journal* 13 (3):329-347.

Fluri, Beat, and Harald C Gall. 2006. "Classifying change types for qualifying change couplings." Program Comprehension, 2006. ICPC 2006. 14th IEEE International Conference on.

Fui-Hoon Nah, Fiona, Janet Lee-Shang Lau, and Jinghua Kuang. 2001. "Critical factors for successful implementation of enterprise systems." *Business process management journal* 7 (3):285-296.

Glass, Robert L. 1998. "Enterprise Resource Planning - Breakthrough and/or Term Problem?" *Data Base* 29 (2):14-16.

Gottschalk, Florian, Wil M. P. van der Aalst, and Monique H. Jansen-Vullers. 2007. "Configurable process models—a foundational approach." In *Reference Modelling*, 59-77. Springer.

Grabski, Severin V, Stewart A Leech, and Pamela J Schmidt. 2011. "A review of ERP research: A future agenda for accounting information systems." *Journal of information systems* 25 (1):37-78.

Guenov, Marin D, and Stephen G Barker. 2005. "Application of axiomatic design and design structure matrix to the decomposition of engineering systems." *Systems engineering* 8 (1):29-40.

Gulledge, Thomas. 2006. "What is integration?" *Industrial Management & Data Systems* 106 (1):5-20.

Gulledge, Thomas, and Georg Simon. 2005. "The evolution of SAP implementation environments: A case study from a complex public sector project." *Industrial Management & Data Systems* 105 (6):714-736.

Ha, Young Mok, and Hyung Jun Ahn. 2014. "Factors affecting the performance of Enterprise Resource Planning (ERP) systems in the post-implementation stage." *Behaviour & Information Technology* 33 (10):1065-1081.

Hamraz, Bahram, Nicholas HM Caldwell, and P John Clarkson. 2012. "A multidomain engineering change propagation model to support uncertainty reduction and risk management in design." *Journal of Mechanical Design* 134 (10):100905.

Harker, Susan DP, Ken D Eason, and John E Dobson. 1993. "The change and evolution of requirements as a challenge to the practice of software engineering." Requirements Engineering, 1993., Proceedings of IEEE International Symposium on.

Hassan, Ahmed E, and Richard C Holt. 2004. "Studying the evolution of software systems using evolutionary code extractors." Software Evolution, 2004. Proceedings. 7th International Workshop on Principles of.

Hevner, Alan R, Salvatore T March, Jinsoo Park, and Sudha Ram. 2004. "Design science in information systems research." *MIS Quarterly* 28 (1):75-105.

Hillman Willis, T., and Ann Hillary Willis-Brown. 2002. "Extending the value of ERP." *Industrial Management & Data Systems* 102 (1):35-38.

Holland, Christopher P., and Ben Light. 1999. "A critical success factors model for ERP implementation." *IEEE software* 16 (3):30.

Hong, Kyung-Kwon, and Young-Gul Kim. 2002. "The critical success factors for ERP implementation: an organisational fit perspective." *Information & Management* 40 (1):25-40.

Hsu, Ko, Jeanne Sylvestre, and Ellen N Sayed. 2006. "Avoiding ERP pitfalls." *Journal of Corporate Accounting & Finance* 17 (4):67-74.

Huang, GQ, and KL Mak. 1998. "Re-engineering the product development process with 'design for X '." *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 212 (4):259-268.

Huang, GQ, WY Yee, and KL Mak. 2001. "Development of a web-based system for engineering change management." *Robotics and Computer-Integrated Manufacturing* 17 (3):255-267.

Huang, Shi-Ming, I. Chu Chang, Shing-Han Li, and Ming-Tong Lin. 2004. "Assessing risk in ERP projects: identify and prioritize the factors." *Industrial management & data systems* 104 (8):681-688.

Hwang, Wonil, and Gavriel Salvendy. 2010. "Number of people required for usability evaluation: the 10±2 rule." *Communications of the ACM* 53 (5):130-133.

Ifinedo, Princely, and Nazmun Nahar. 2006. "Quality, impact and success of ERP systems: a study involving some firms in the Nordic-Baltic region." *Journal of Information Technology Impact* 6 (1):19-46.

Ifinedo, Princely, and Nazmun Nahar. 2007. "ERP systems success: an empirical analysis of how two organisational stakeholder groups prioritize and evaluate relevant measures." *Enterprise Information Systems* 1 (1):25-48.

Ifinedo, Princely, Birger Rapp, Airi Ifinedo, and Klas Sundberg. 2010. "Relationships among ERP post-implementation success constructs: An analysis at the organisational level." *Computers in Human Behaviour* 26 (5):1136-1148.

Iivari, Juhani. 1991. "A paradigmatic analysis of contemporary schools of IS development." *European Journal of Information Systems* 1 (4):249-272.

Inness, John G. 1994. *Achieving successful product change: A handbook*: Financial Times/Pitman.

ISO-9126. 2010. Systems and software engineering in *Systems and software Quality and Requirements and Evaluation*: ISO/ IEC.

Jacobson, Ivar, Grady Booch, James Rumbaugh, James Rumbaugh, and Grady Booch. 1999. *The unified software development process*. Vol. 1: Addison-Wesley Reading.

Jarratt, TAW, Claudia M Eckert, NHM Caldwell, and P John Clarkson. 2011. "Engineering change: an overview and perspective on the literature." *Research in engineering design* 22 (2):103-124.

Jarratt, Timothy AW. 2004. "A model-based approach to support the management of engineering change." Cambridge University Engineering Department.

Johannesson, Paul, and Erik Perjons. 2001. "Design principles for process modelling in enterprise application integration." *information systems* 26 (3):165-184.
Jönsson, Per. 2007. "Exploring Process Aspects of Change Impact Analysis."
Kama, Nazri. 2013. "Change impact analysis for the software development phase: state-of-the-art." *International Journal of Software Engineering and Its Application* 7 (2).
Kang, Jaewoo, and Jeffrey F Naughton. 2003. "On schema matching with opaque column names and data values." Proceedings of the 2003 ACM SIGMOD international conference on Management of data.
Keil, Mark, and Amrit Tiwana. 2005. "Beyond cost: the drivers of COTS application value." *Software, IEEE* 22 (3):64-69.
Kherbouche, Oussama Mohammed, Ayaz Ahmad, Mourad Bouneffa, and Henri Basson. 2013. "Analysing the ripple effects of change in business process models." Multi Topic Conference (INMIC), 2013 16th International.
Kilpinen, Malia Sofia. 2008. "The Emergence of Change at the Systems Engineering and Software Design Interface." University of Cambridge.
Klaus, Helmut, Michael Rosemann, and Guy G Gable. 2000. "What is ERP?" *Information systems frontiers* 2 (2):141-162.
Kruchten, Philippe. 2004. *The rational unified process: an introduction*: Addison-Wesley Professional.
Kumar, Kuldeep, and Jos van Hillegersberg. 2000. "Enterprise resource planning: introduction." *Communications of the ACM* 43 (4):22-26.
Kung, D, Jerry Gao, Pei Hsia, F Wen, Yasufumi Toyoshima, and Cris Chen. 1994. "Change impact identification in object oriented software maintenance." Software Maintenance, 1994. Proceedings., International Conference on.
Kurbel, Karl E. 2013. Enterprise Resource Planning and Supply Chain Management. Springer.
La Rosa, Marcello, Marlon Dumas, Arthur H. M. Ter Hofstede, and Jan Mendling. 2011. "Configurable multi-perspective business process models." *Information Systems* 36 (2):313-340.
Langermeier, Melanie, Christian Saad, and Bernhard Bauer. 2014. "Adaptive Approach for Impact Analysis in Enterprise Architectures." In *Business Modelling and Software Design*, 22-42. Springer.
Laudon, Kenneth C, and Jane Price Laudon. 2000. *Management information systems*. Vol. 6: Prentice Hall Upper Saddle River, NJ.
Law, Chuck CH, Charlie C Chen, and Bruce JP Wu. 2010. "Managing the full ERP life-cycle: Considerations of maintenance and support requirements and IT governance practice as integral elements of the formula for successful ERP adoption." *Computers in Industry* 61 (3):297-308.
Leau, Yu Beng, Wooi Khong Loo, Wai Yip Tham, and Soo Fun Tan. 2012. "Software development life cycle AGILE vs traditional approaches." International Conference on Information and Network Technology.
Lechesa, Motheo, Lisa Seymour, and Joachim Schuler. 2012. "ERP Software as Service (SaaS): factors affecting adoption in South Africa." In *Re-conceptualizing Enterprise Information Systems*, 152-167. Springer.
Lee, Michelle L. 1998. "Change impact analysis of object-oriented software." Citeseer.
Leech, DJ, and Barry Trevor Turner. 1985. *Engineering design for profit*: Halsted Press.
Lehnert, Steffen. 2011a. "A review of software change impact analysis." *Ilmenau University of Technology, Tech. Rep*.
Lehnert, Steffen. 2011b. "A taxonomy for software change impact analysis." 2011.

Lehnert, Steffen, and Matthias Riebisch. 2012. "A taxonomy of change types and its application in software evolution." Engineering of Computer Based Systems (ECBS), 2012 IEEE 19th International Conference and Workshops on.

Leon, Alexis. 2000. *A Guide to software configuration management*: Artech House, Inc.

Leyh, Christian. 2011. "Tailoring of ERP Systems." *Readings on Enterprise Resource Planning (Preliminary Version). Montreal: ERsim Lab, HEC Montreal*.

Light, Ben. 2001. "The maintenance implications of the customization of ERP software." *Journal of software maintenance and evolution: research and practice* 13 (6):415-429.

Light, Ben. 2005. "Going beyond 'misfit'as a reason for ERP package customisation." *Computers in Industry* 56 (6):606-619.

Lindvall, Mikael. 1998. "Monitoring and measuring the change-prediction process at different granularity levels." *Software Process: Improvement and Practice* 4 (1):3-10.

Lorenz, Mark, and Jeff Kidd. 1994. *Object-oriented software metrics: a practical guide*: Prentice-Hall, Inc.

Lübke, Daniel, and Jorge Marx Gómez. 2010. "Developing and Customizing Federated ERP Systems."

Lucas Jr, Henry C, Eric J Walton, and Michael J Ginzberg. 1987. "Implementing packaged software." *Information Systems Working Papers Series, Vol*.

Lucas, Wendy, and Tamara Babaian. 2012. "Implementing design principles for collaborative ERP systems." International Conference on Design Science Research in Information Systems.

Lund, Arnold M. 2001. "Measuring Usability with the USE Questionnaire12."

Luo, Wenhong, and Diane M. Strong. 2004. "A framework for evaluating ERP implementation choices." *Engineering Management, IEEE Transactions on* 51 (3):322-333.

Mafazi, Shamila, Georg Grossmann, Wolfgang Mayer, and Markus Stumptner. 2013. "On-the-fly change propagation for the co-evolution of business processes." On the Move to Meaningful Internet Systems: OTM 2013 Conferences.

Manuel, Paul D, and Jarallah AlGhamdi. 2003. "A data-centric design for n-tier architecture." *Information Sciences* 150 (3):195-206.

Markus, M. Lynne, David Petrie, and Sheryl Axline. 2003. "18 Continuity Versus Discontinuity: Weighing the Future of ERP Packages." *Second-Wave Enterprise Resource Planning Systems: Implementing for Effectiveness*:9.

Markus, M. Lynne, and Cornelis Tanis. 2000. "The enterprise systems experience-from adoption to success." *Framing the domains of IT research: Glimpsing the future through the past* 173:207-173.

Mento, Anthony, Raymond Jones, and Walter Dirndorfer. 2002. "A change management process: Grounded in both theory and practice." *Journal of Change Management* 3 (1):45-59.

Milanović, M. 2007. "Modelling rules on the Semantic Web." Master thesis, University of Belgrade.

Miller, Stewart S. 1998. *Accelerated Sap (ASAP): Implementation at the Speed of Business*: McGraw-Hill, Inc.

Moalagh, Morteza, and Ahad Zare Ravasan. 2013. "Developing a practical framework for assessing ERP post-implementation success using fuzzy analytic network process." *International Journal of Production Research* 51 (4):1236-1257.

Munkelt, Torsten. 2013. "ERP systems: aspects of selection, implementation and sustainable operations."

Muscatello, Joseph R, and Injazz J Chen. 2008. "Enterprise resource planning (ERP) implementations: theory and practice." *International Journal of Enterprise Information Systems* 4 (1):63.

Nah, Fiona Fui-Hoon, Kathryn M. Zuckweiler, and Janet Lee-Shang Lau. 2003. "ERP implementation: chief information officers' perceptions of critical success factors." *International Journal of Human-Computer Interaction* 16 (1):5-22.

NAV, Microsoft Dynamics. 2016. "CRONUS International Ltd. Demonstration Database." Msdn.microsoft.com Accessed November 2015. https://msdn.microsoft.com/en-us/library/hh997462%28v=nav.90%29.aspx.

Netweaver, SAP. 2009. Germany.

Ng, Celeste See Pui, Guy G Gable, and Taizan Chan. 2002. "An ERP-client benefit-oriented maintenance taxonomy." *Journal of Systems and Software* 64 (2):87-109.

Ng, Celeste See-Pui. 2012. "A case on ERP custom add-on in Taiwan: implications to system fit, acceptance and maintenance costs." *International Journal of Enterprise Information Systems (IJEIS)* 8 (4):44-62.

O'Brien, James A, and George M Marakas. 2006. *Enterprise information systems*: McGraw-Hill Irwin.

Oates, Briony J. 2005. *Researching information systems and computing*: Sage.

Oliva, Gustavo A, Marco A Gerosa, Dejan Milojicic, and Valton Smith. 2013. "A change impact analysis approach for workflow repository management." Web Services (ICWS), 2013 IEEE 20th International Conference on.

Ollinger, Gabriel Aguirre, and Thomas F Stahovich. 2001. "'RedesignIT—A Constraint-based Tool for Managing Design Changes." Proc. ASME International Design Engineering Technical Conferences, Design Theory and Methodology.

Ollinger, Gabriel Aguirre, and Thomas F Stahovich. 2004. "RedesignIT—a model-based tool for managing design changes." *Journal of Mechanical Design* 126 (2):208-216.

Orlikowski, Wanda J. 2002. "Knowing in practice: Enacting a collective capability in distributed organizing." *Organisation science* 13 (3):249-273.

Oseni, Taiwo, Mahbubur Md Rahim, Stephen P Smith, and Susan Foster. 2014. "An initial empirical evaluation of the influence of erp post-implementation modifications on business process optimisation."

Pajk, Dejan, Mojca Indihar Štemberger, and Andrej Kovačić. 2010. "The Use of Reference Models in Business Process Renovation." *Business Systems Research* 1 (1-2):30-38.

Pajk, Dejan, and Andrej Kovacic. 2013. "FIT GAP ANALYSIS-THE ROLE OF BUSINESS PROCESS REFERENCE MODELS." *Economic and Business Review for Central and South-Eastern Europe* 15 (4):319.

Palomino Murcia, MA, and Edgar a Whitley. 2007. "The effects of national culture on ERP implementation: a study of Colombia and Switzerland." *Enterprise Information Systems* 1 (3):301-325.

Parhizkar, Minou, and Marco Comuzzi. 2015. "An AHP-Based analysis of the Cost of ERP Modification." Enterprise Systems Conference (ES), 2015.

Parr, Anne, and Graeme Shanks. 2000. "A model of ERP project implementation." *Journal of information Technology* 15 (4):289-303.

Parthasarathy, Sudhaman, and Maya Daneva. 2014. "Customer requirements based ERP customization using AHP technique." *Business Process Management Journal* 20 (5):730-751.

Peffers, Ken, Tuure Tuunanen, Marcus A Rothenberger, and Samir Chatterjee. 2007. "A design science research methodology for information systems research." *Journal of management information systems* 24 (3):45-77.

Pianwattanaphon, Rewat, and Twittie Senivongse. 2007. "Compatibility by Service Type Model for Automatic Web Services Substitution." Advanced Communication Technology, The 9th International Conference on.

Pikosz, Peter, and Johan Malmqvist. 1998. "A comparative study of engineering change management in three Swedish engineering companies." Proceedings of the DETC98 ASME design engineering technical conference.

Queille, Jean-plene, Jean-Franmis Voidrot, Norman Wilde, and Malcolm Munro. 1994. "The impact analysis task in software maintenance: a model and a case study." Software Maintenance, 1994. Proceedings., International Conference on.

Qumer, Asif, and Brian Henderson-Sellers. 2008. "A framework to support the evaluation, adoption and improvement of agile methods in practice." *Journal of Systems and Software* 81 (11):1899-1919.

Rajlich, Vaclav, and Prashant Gosavi. 2004. "Incremental change in object-oriented programming." *Software, IEEE* 21 (4):62-69.

Ram, Jiwat, David Corkindale, and Ming-Lu Wu. 2013. "Implementation critical success factors (CSFs) for ERP: Do they contribute to implementation success and post-implementation performance?" *International Journal of Production Economics* 144 (1):157-174.

Rashid, Mohammad A, Liaquat Hossain, and Jon David Patrick. 2002. The evolution of ERP systems: A historical perspective.

Reichert, Manfred, and Peter Dadam. 1998. "ADEPTflex—Supporting dynamic changes of workflows without losing control." *Journal of Intelligent Information Systems* 10 (2):93-129.

Reichert, Manfred, and Barbara Weber. 2012. *Enabling flexibility in process-aware information systems: challenges, methods, technologies*: Springer Science & Business Media.

Rodrigo. 2012. "Evaluation of Mendix as Rapid Application Development (RAD) tool." http://writepass.com/journal/2012/12/evaluation-of-mendix-as-a-rapid-application-development-rad-tool/.

Rohatgi, Abhishek, Abdelwahab Hamou-Lhadj, and Juergen Rilling. 2008. "An approach for mapping features to code based on static and dynamic analysis." Program Comprehension, 2008. ICPC 2008. The 16th IEEE International Conference on.

Rohatgi, Ajeet, Abdelwahab Hamou-Lhadj, and Juergen Rilling. 2009. "Approach for solving the feature location problem by measuring the component modification impact." *Software, IET* 3 (4):292-311.

Rothenberger, Marcus A, and Mark Srite. 2009. "An investigation of customization in ERP system implementations." *Engineering Management, IEEE Transactions on* 56 (4):663-676.

Saaty, Thomas L. 2003. "Decision-making with the AHP: Why is the principal eigenvector necessary." *European journal of operational research* 145 (1):85-91.

Saaty, Thomas L, and Luis G Vargas. 2012. *Models, methods, concepts & applications of the analytic hierarchy process*. Vol. 175: Springer Science & Business Media.

Salmeron, Jose L., and Cristina Lopez. 2012. "Forecasting risk impact on ERP maintenance with augmented fuzzy cognitive maps." *Software Engineering, IEEE Transactions on* 38 (2):439-452.

Sarfaraz, Ahmad, Kouroush Jenab, and Amanda C D'Souza. 2012. "Evaluating ERP implementation choices on the basis of customisation using fuzzy AHP." *International Journal of Production Research* 50 (23):7057-7067.

Scott, Judy E, and Lisa Kaindl. 2000. "Enhancing functionality in an enterprise software package." *Information & Management* 37 (3):111-122.

Selmeci, Attila, István Orosz, Gy Györök, and Tamás Orosz. 2012. "Key Performance Indicators used in ERP performance measurement applications." Intelligent Systems and Informatics (SISY), 2012 IEEE 10th Jubilee International Symposium on.

Shang, Shari, and Peter B Seddon. 2002. "Assessing and managing the benefits of enterprise systems: the business manager's perspective." *Information systems journal* 12 (4):271-299.

Shaw, Mary. 2002. "What makes good research in software engineering?" *International Journal on Software Tools for Technology Transfer* 4 (1):1-7.

Shiang-Yen, Tan, Rosnah Idrus, and Wai Peng Wong. 2013. "ERP Misfit-Reduction Strategies: A Moderated Model of System Modification and Organisational Adaptation." *Journal of Global Information Management (JGIM)* 21 (1):59-81.

Smets-Solanes, Jean-Paul, and Rogério Atem De Carvalho. 2003. "ERP5: a next-generation, open-source ERP architecture." *IT professional* 5 (4):38-44.

Soffer, Pnina, Boaz Golany, and Dov Dori. 2003. "ERP Modelling: a comprehensive approach." *Information systems* 28 (6):673-690.

Soh, Christina, Sia Siew Kien, and Joanne Tay-Yap. 2000. "Enterprise resource planning: cultural fits and misfits: is ERP a universal solution?" *Communications of the ACM* 43 (4):47-51.

Soja, P. 2009. "Enterprise system implementation issues: learning from field study in Poland." *Enterprise Information Systems* 3 (2):173-200.

Soley, Richard. 2000. "Model driven architecture." *OMG white paper* 308 (308):5.

Somers, Toni M, and Klara Nelson. 2001. "The impact of critical success factors across the stages of enterprise resource planning implementations." System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on.

Somers, Toni M, and Klara G Nelson. 2004. "A taxonomy of players and activities across the ERP project life cycle." *Information & Management* 41 (3):257-278.

Somers, Toni M., and Klara G. Nelson. 2003. "The impact of strategy and integration mechanisms on enterprise system value: Empirical evidence from manufacturing firms." *European Journal of Operational Research* 146 (2):315-338.

Štemberger, Mojca Indihar, and Andrej Kovačič. 2008. "The role of business process modelling in ERP implementation projects." Computer Modelling and Simulation, 2008. UKSIM 2008. Tenth International Conference on.

Sumner, Mary. 2000. "Risk factors in enterprise-wide/ERP projects." *Journal of information technology* 15 (4):317-327.

Sun, Albert YT, Abe Yazdani, and John D Overend. 2005. "Achievement assessment for enterprise resource planning (ERP) system implementations based on critical success factors (CSFs)." *International journal of production economics* 98 (2):189-203.

Sun, Ping, and Changjun Jiang. 2009. "Analysis of workflow dynamic changes based on Petri net." *Information and Software Technology* 51 (2):284-292.

Sun, Xiaobing, Bixin Li, Chuanqi Tao, Wanzhi Wen, and Sai Zhang. 2010. "Change impact analysis based on a taxonomy of change types." Computer Software and Applications Conference (COMPSAC), 2010 IEEE 34th Annual.

Sunkle, Sagar, Vinay Kulkarni, and Suman Roychoudhury. 2013. "Analysable Enterprise Models Using Ontology." CAiSE Forum.

Tarn, J Michael, David C Yen, and Marcus Beaumont. 2002. "Exploring the rationales for ERP and SCM integration." *Industrial Management & Data Systems* 102 (1):26-34.

Terwiesch, Christian, and Christoph H Loch. 1999. "Managing the process of engineering change orders: the case of the climate control system in automobile development." *Journal of Product Innovation Management* 16 (2):160-172.

Themistocleous, Marinos, Zahir Irani, Robert M O'Keefe, and Ray Paul. 2001. "ERP problems and application integration issues: An empirical survey." System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on.

Umble, Elisabeth J, Ronald R Haft, and M Michael Umble. 2003. "Enterprise resource planning: Implementation procedures and critical success factors." *European journal of operational research* 146 (2):241-257.

Upadhyay, Parijat, Saeed Jahanyan, and Pranab K Dan. 2011. "Factors influencing ERP implementation in Indian manufacturing organisations: A study of micro, small and medium-scale enterprises." *Journal of Enterprise Information Management* 24 (2):130-145.

Uppstrom, Elin, Carl-Mikael Lonn, Madeleine Hoffsten, and Joakim Thorstrom. 2015. "New Implications for Customization of ERP Systems." System Sciences (HICSS), 2015 48th Hawaii International Conference on.

van Beijsterveld, Joost. 2006. "Misfits in ERP System Implementation." Master thesis.

Vernadat, François B. 2003. *Enterprise modelling and integration*: Springer.

Von Knethen, Antje. 2002. "Change-oriented requirements traceability. Support for evolution of embedded systems." Software Maintenance, 2002. Proceedings. International Conference on.

Wang, Yi, Jian Yang, and Weiliang Zhao. 2010. "Change impact analysis for service based business processes." Service-Oriented Computing and Applications (SOCA), 2010 IEEE International Conference on.

Wang, Zhong-Jie, and Xiao-Fei Xu. 2009. "Business process integration point classification and the priority evaluation method." *International Journal of Business Information Systems* 4 (2):210-232.

Weber, Barbara, Stefanie Rinderle, and Manfred Reichert. 2007a. "Change patterns and change support features in process-aware information systems." Advanced Information Systems Engineering.

Weber, Barbara, Stefanie Rinderle, and Manfred Reichert. 2007b. "Change patterns and change support features in process-aware information systems." In *Advanced Information Systems Engineering*, 574-588. Springer.

Weber, Barbara, Shazia Sadiq, and Manfred Reichert. 2009. "Beyond rigidity–dynamic process lifecycle support." *Computer Science-Research and Development* 23 (2):47-65.

Wei, Hsiao-Lan, Eric TG Wang, and Pei-Hung Ju. 2005. "Understanding misalignment and cascading change of ERP implementation: a stage view of process analysis." *European Journal of Information Systems* 14 (4):324-334.

Weske, Mathias. 1998. "Flexible Modelling and execution of workflow activities." System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on.

Wieringa, Roel J. 2014. *Design science methodology for information systems and software engineering*: Springer.

Winter, Robert. "Design science research in Europe." *European Journal of Information Systems* 17, no. 5 (2008): 470-475.

Winter, Robert, and Ronny Fischer. 2006. "Essential layers, artefacts, and dependencies of enterprise architecture." Enterprise Distributed Object Computing Conference Workshops, 2006. EDOCW'06. 10th IEEE International.

Wright, IC. 1997. "A review of research into engineering change management: implications for product design." *Design Studies* 18 (1):33-42.

Wu, Jian, and Zhaohui Wu. 2005. "Similarity-based web service matchmaking." Services Computing, 2005 IEEE International Conference on.

Xiao, Hua, Jin Quo, and Ying Zou. 2007. "Supporting change impact analysis for service oriented business applications." Systems Development in SOA Environments, 2007. SDSOA'07: ICSE Workshops 2007. International Workshop on.

Yakovlev, Ilya V. 2002. "An ERP Implementation and Business Process Reengineering at a Small University." *Educause Quarterly* 25 (2):52-57.

Yan, YuQing, Shixian Li, and Weijun Sun. 2012. "Dependency based Technique for Identifying the Ripple Effect of Requirements Evolution." *Journal of Software* 7 (3):544-550.

Yan, Zhiqiang, Remco Dijkman, and Paul Grefen. 2010. "Fast business process similarity search with feature-based similarity estimation." In *On the Move to Meaningful Internet Systems: OTM 2010*, 60-77. Springer.

Yen, Tan Shiang, Rosnah Idrus, and Umi Kalsom Yusof. 2011. "A Framework for classifying misfits between enterprise resource planning (ERP) systems and business strategies." *Asian Academy of Management Journal* 16 (2):53-75.

Yu, Chian-Son. 2005. "Causes influencing the effectiveness of the post-implementation ERP system." *Industrial Management & Data Systems* 105 (1):115-132.

Zach, Ondrej, and Bjorn Erik Munkvold. 2012. "Identifying reasons for ERP system customization in SMEs: a multiple case study." *Journal of Enterprise Information Management* 25 (5):462-478.

Zhao, Jianjun, Hongji Yang, Liming Xiang, and Baowen Xu. 2002. "Change impact analysis to support architectural evolution." *Journal of software maintenance and evolution: research and practice* 14 (5):317-333.

Zhao, Xiaohui, and Chengfei Liu. 2013. "Version management for business process schema evolution." *Information Systems* 38 (8):1046-1069.

Zhao, Xulin, and Ying Zou. 2011. "A business process- driven approach for generating software modules." *Software: Practice and Experience* 41 (10):1049-1071.

Zhu, Yan, Yan Li, Weiquan Wang, and Jian Chen. 2010. "What leads to post-implementation success of ERP? An empirical study of the Chinese retail industry." *International Journal of Information Management* 30 (3):265-276.

APPENDIX. A IMPACT ALGORITHM

Algorithm 10: Updating Business Function:

Input: UpdateFunction (f_i, f_i')
begin
    COMPATIBLE ← CompatibilityCheck (f_i, f_i')
    if COMPATIBLE = TRUE **then**
        CREATE f_i' //Change can be implemented
        PROC ⊆ PRO ← $\{pro_p: Contain\ (pro_p, f_i)\}$
        for each pro_p ∈ PROC **Do**
            INST ⊆ INS ← $\{ins_{p,j}: Create\ (pro_p, ins_{p,j})\}$
            Migrate All ins ∈ INST_{Pro} to f_i'
        end for
        Delete f_i
    else // COMPATIBLE = FALSE
        CREATE f_i'
            PROC ⊆ PRO ← $\{pro_p: Uses\ (pro_p, f_i)\}$
            for each pro_p ∈ PROC **do**
                INTG_{PRO} ⊆ PRO ← $\{pro_p: Called\ by\ (pro_p, pro_q)\}$
                for each pro ∈ INTG_{PRO} **do**
                    Call $\boldsymbol{INTG_{PRO}}$
                end for
                INS ⊆ INS_p ← $\{ins: Create\ (pro_p, ins)\}$
                $\text{INS}_{with\ INTG}$ ⊆ INS ← $\{ins: Create\ (ins_{p,b}, ins_{q,a})\}$
                for each ins ∈ INS **do**
                    if ins ∈ $\text{INS}_{with\ INTG}$ **then**
                        CP ← calculateCriticalPoint (pro_p, pro_q)
                    else
                        CP ← calculateCriticalPoint (f_i, pro_p)
                    end if
                    PASSED_CP ← checkCriticalPoint (ins, CP)
                    if PASSED_CP **then**
                        Migrate
                    else
                        Flush then Migrate
                    end if
                end for
            end for
        end for
        Delete f_i
    end if
end

Algorithm 11: Updating Business Process:

Input: UpdateProcess $(pro_p, pro_p{}')$
begin
  COMPATIBLE $\leftarrow$ CompatibilityCheck $(pro_p, pro_p{}')$
  if COMPATIBLE = TRUE **Then**
    CREATE $pro_p{}'$
    $\text{INST}_{\text{Pro}} \subseteq \text{INS} \leftarrow \{ins: Create\ (pro_p, ins)\}$
    Migrate All $ins \in \text{INST}_{\text{Pro}}$ to $pro_p{}'$
    Delete pro_p
  else // COMPATIBLE = FALSE
    CREATE $pro_p{}'$
    $\text{INTG}_{\text{PRO}} \subseteq \text{PRO} \leftarrow \{pro_p: Called\ by\ (pro_p, pro_q)\}$
    for each $pro \in \text{INTG}_{\text{PRO}}$ **do**
      Call $\boldsymbol{INTG_{PRO}}$
    end for
    $\text{INS} \subseteq \text{INS}_p \leftarrow \{ins: Create\ (pro_p, ins)\}$
    $\text{INS}_{with\ INTG} \subseteq \text{INS} \leftarrow \{ins: Create\ (ins_{p,b}, ins_{q,a})\}$
    for each $ins \in \text{INS}$ **do**
      if $ins \in \text{INS}_{with\ INTG}$ **then**
        CP $\leftarrow$ calculateCriticalPoint (pro_p, pro_q)
      else
        CP $\leftarrow$ calculateCriticalPoint (σ, pro_p)
      end if
      PASSED_CP $\leftarrow$ checkCriticalPoint (ins, CP)
      if PASSED_CP **then**
        Migrate
      else
        Flush then Migrate
      end if
    end for
    Delete pro_p
  end if
end

Algorithm 12: $\mathbf{INTG_{PRO}}$ (Process Integration)

Input: INTG_{PRO} (pro_p, PRO)
begin
    $\text{INTG}_{\text{PRO}} \subseteq \text{PRO} \leftarrow \{pro_p: Called\ by\ (pro_p, pro_q)\}$
    for each $pro \in \text{INTG}_{\text{PRO}}$ **do**
        call $\boldsymbol{INTG_{PRO}}$
    end for
    $\text{INS} \subseteq \text{INS}_p \leftarrow \{ins: Create\ (pro_p, ins)\}$
    $\text{INS}_{with\ INTG} \subseteq \text{INS} \leftarrow \{ins: Create\ (\ ins_{p,b}\ , ins_{q,a})\}$
    for each $ins \in \text{INS}$ **do**
        if $ins \in \text{INS}_{with\ INTG}$ **then**
            CP ← calculateCriticalPoint (pro_p, pro_q)
        else
            CP ← calculateCriticalPoint (σ, pro_p)
        end if
        PASSED_CP ← checkCriticalPoint (ins, CP)
        if PASSED_CP **then**
            Migrate
        else
            Flush then Migrate
        end if;
    end for
end

Algorithm 13: Deleting Business Data

Input DeleteData $(d_k \in D)$
begin
    $\text{FUNC} \subseteq \text{F} \leftarrow \{f_i: Interact\ (f_i, d_k)\}$
    if FUNC = EMPTY **then**
        DELETE
    else
        for each $f_i \in \text{FUNC}$ **do**
            $\text{PROC} \subseteq \text{PRO} \leftarrow \{pro_p: Contain\ (pro_p, f_i)\}$
            if PROC=EMPTY **then**
                Check delete f_i
            else
                Print PROC
            end
        end
        Print FUNC
    end
end

Algorithm 14: Deleting Business Function:

Input DeleteFuntion ($f_i \in F$)
begin
    PROC ⊆ PRO ← $\{pro_p: Contain\ (pro_p, f_i)\}$
    if PROC = EMPTY **then**
        FUNC_OUT ← $\{f_i: Interact\ (f_i, d_k, O)\}$ //the processes that are impacted
        if FUNC_OUT = EMPTY **then**
            DELETE
        else
            for each d_k ∈ FUNC_OUT **do**
                FUNC ⊆ F ← $\{f_i: Interact\ (f_i, d_k)\}$
                if FUNC = EMPTY **then**
                    Check delete d_k
                else
                    Print FUNC
                end
            end
            Print FUNC_OUT
        end
    else
        Print PROC
    end
end

Algorithm 15: Deleting Business Process:

Input DeleteProcess ($pro_p \in PRO$)
begin
    INT_{PRO} ⊆ PRO ← $pro: Integrate\ (pro_p, pro)\}$ //Process integration
    if INT_{PRO} = EMPTY **then**
    INST ⊆ INS ← $\{ins_{p,j}: Create\ (pro_p, ins_{p,j})\}$
    for each $ins_{p,j}$ ∈ INST **do**
        if |INST| < THRESHOLD **then**
            //Threshold defines at the requirement level, the number of active instances to be change
            Abort
        else
            Flush
        end
    end
    else
        Print INT_{PRO}
    end
end

APPENDIX. B QUESTIONNAIRE FIRST STUDY

WELCOME PAGE

Dear Participant,

My name is Minou Parhizkar, and I am a PhD Student with the Department of Computing at City University London, completing my thesis on the topic *Impact Analysis of on-demand ERP modifications.*

I require your expertise on ERP to fill in survey that will help me in the analysis of the "costs" of on-demand ERP modifications. This is calibrated to take no more than 8 minutes of your time.

Structure of Survey:
The survey is organized in 4 sections. Each section focuses on a specific level at which modifications of ERP systems may be required:

- Function level, e.g. modifying the way a purchase order is created or updated
- Business process level, e.g. modifying the way a purchase order is completed by different departments
- Data level (and documents), e.g. modifying the attributes and/or structure of the purchase order business object
- Overall comparison of different types of ERP modifications

In each section, you will be asked to compare "pairwise" the costs of different types of modification that are applicable at a given level.
Click on the right-bottom corner to proceed
Q1 Please specify your age:
Q2 Please select your current occupation.

- ❑ Project Manager (ISO-9126)
- ❑ Senior Consulting Manager (ISO-9126)
- ❑ Consultant/Advisor (3)
- ❑ Developer (4)
- ❑ Business Change Manager (5)
- ❑ Business System Analyst (7)
- ❑ IT Specialist (8)
- ❑ Others (ISO-9126) _____

Q3 For how long do you have working experience in ERP system?

A) FUNCTION LEVEL

Types of ERP Function Modification to compare:

1) Configuration: involves selecting from the ERP reference model and changing the setting of parameters in order to choose between different executions of processes and functions in the software package.

2) Bolt-on (add-on): Utilizing third party solution designed to work seamlessly with the ERP package in order to supplement the specific functionality required

3) User Exits: Type of modification that requires software programming where users can arrange for tailor made code to extend the functionality of a given ERP package. This is limited to specific "functions" that are pre-defined by the ERP vendor (i.e. Develop a statistical function for calculating particular metrics)

4) ERP programming: This type of modification involves the extension of the ERP package using the standard language in which the ERP package is developed, e.g. ABAP in SAP. (I.e. programming additional application without changing ERP system code)

5) Interface Development: This type of modification involves using specific technology (e.g. Web services) to bridge the gap between the ERP package and other systems, e.g. in the case of database-to-database interfacing or interfaces to legacy systems.

6) Code modification: This refers to modification of the ERP package source code

Scaling

3 = Slightly (Cheaper/Expensive)
5 = Moderately (Cheaper/Expensive)
7 = Extremely (Cheaper/Expensive)
9 = Very Extremely (Cheaper/Expensive)
The number 2***,4,6,8*** used as intermediate values between above Scaling

Use ***1 is*** to Capture that two types of modifications are equally expensive

Example

If you want to capture that "Bolt-on type of modification for ERP functions is ***moderately*** more expensive than Configuration" **then you should tick the radio button "expensive" and choose the scale 5 in the first line of the matrix below.**

Q1) Please Compare Rows with Column for each Type of Modifications at Functional Level
For example: "Bolt-on type of modification for ERP functions is moderately more expensive than Configuration"

	.... Than Configuration		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Bolt-on is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The User-Exits is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The ERP-Programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Code modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Bolt-on		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The User-Exits is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The ERP-Programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Code modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than User-Exits		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The ERP-Programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Code modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than ERP-Programming		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Code modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Interface Development		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Code modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

B) Business Process Level

Type of ERP Business Process modifications to compare:

1) Configuration: involves selecting from the ERP reference model and changing the setting of parameters in order to choose between different executions of processes and functions in the software package, e.g. configuring different policies for inventory management.

2) Bolt-on (add-on): Utilizing business processes provided by a third party and that can work seamlessly with the ERP package in order to supplement the specific the processes required

3) Work-flow Programming: This concerns the modification of the ERP standard workflows (e.g. EPC process models in SAP) in case they are not sufficient to fulfil the needs of the user (i.e. adding intermediate work-flow state to support more complex decision processes of an organisation)

4) ERP programming: This type of modification involves the extension of the ERP package using the standard language in which the ERP package is developed, e.g. ABAP in SAP. (I.e. developing a new embedded business process without changing the ERP system code)

5) Interface Development: This type of modification involves using specific technology (e.g. Web services) to bridge the gap between the ERP package and other systems for creating new processes addressing the need for change

Q1 Please Compare Rows with Column for each Type of Modifications.

	.... Than Configuration		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Bolt-on is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The workflow programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The ERP-Programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Bolt-on		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The workflow programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The ERP-Programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Workflow Programming		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The ERP-Programming is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than ERP-Programming		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

C) Data Level

1) Configuration: involves selecting from the ERP reference model and changing the setting of parameters in order to choose between different executions of processes and functions in the software package, e.g. configuring different attributes of a purchase requisition.

***2) Interface Development*:** This type of modification involves using specific technology (e.g. Web services) to bridge the gap between the ERP package and other systems for accessing data (i.e. business objects) that are relevant for the identified change.

3) Query Modification: Direct modification of ERP database entities and relationship.

Q1 Please Compare Rows with Column for each Type of Modifications.

	.... Than Configuration		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Interface Development is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Query Modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Interface Development		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Query Modification is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

D) Overall Comparison

Please do compare Two different type of Modifications (Configuration and Code Modification) by different level of Change (Data, Function, Process)

Example:
Compare ***Configuration of Data*** with ***Configuration of Function*** and define which one is Cheaper in your opinion

Q1 Please Compare Rows with Column for each Type of Modifications.

	.... Than Configuration of Process		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Configuration of Data is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Configuration of Function is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Configuration of Data		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Configuration of Function is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Query Modification Data		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Code Modification (Function) is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
The Workflow Programming (Process) is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

	.... Than Code Modification		Scale from 1-9								
	Expensive	Cheaper	1	2	3	4	5	6	7	8	9
The Workflow Programming (Process) is	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

APPENDIX. C QUESTIONNAIRE SECOND STUDY

Part A) Study with ERP Expert

Session I. Investigation on the current approach

Please provide following information

Occupation _____

Field of expertise _____

Please provide years of experience in ERP system _____

Interview Current Approach

1. Do you have any experience using impact analysis tool in your ERP system? (e.g. Regression testing, SAP Solution Manager 7.0, PanayaIA, etc.)

   _____

2. How do you analyse change propagation in your ERP system? Do you use any informal way (e.g. expert judgment, previous historical data, and previous experience, etc.) or any other formal method or technique?

   _____

3. Do you have a standard set of steps/activities or a process to trace the impact of modification in your ERP system?

   _____

4. What kind of information is available in relation to impact analysis and what information do you use during the analysis process? (I.e. traceability/dependency model, change documentation, historical change data, etc.)

   _____

5. How important for you to use a structured impact analysis approach when assessing the impact of a modification in your ERP system?

_____

Session II. Demonstration of the impact analysis tool

A) Explore ERP system features and functionalities of the Case study

In this session, we are providing an overview of an ERP system case study as an instantiation of the system. A case study is GBI a bike manufacturing company that implement SAP ERP in order to manage and run the activities during the process of manufacturing of Bike. GBI* Case Study demonstrates the fundamental business processes interact with SAP ERP in the area of Sales and Distribution, Material Management and Production Planning. We map the business process, functions/activities, and documents/data object from this case study into our impact analysis tool in order to identify the dependency model. This information will be used for testing purposes to define the effects of change on this particular case study.

> **Global Bike Inc. has a pragmatic design philosophy that comes from its deep roots in both the off-road trail racing and long-distance road racing sports. They manufacturing bike, selling to their customer and procuring semi-finished and raw material from suppliers.*

B) Process of assessing the ERP modification

In this session, we provide you with the guidelines, before presenting the impact analysis demo. The template in the following table captures the basic requirement to proceed with the change request before implementing in ERP systems. This template is used as a guideline for assessing the changing request by the business analyst to define and prioritizing the change request.

Please take some time to read the guidelines before running any examples

Change Request template

Change Request Guideline	
Element	***Description***
CR-Code	The exclusive Code When Change Request was created
Date Reported	The date the Change Request was created
Requested by	Assigned by the ERP end-user (Consultant)
Title	A brief description of the change request
Submitter	Name of the person completing the CR Form and who can answer questions regarding the suggested change
Description	Description of the desired change and how the change Should works
Primitive Change	A list of basic requirements (‘i.e. Primitive Change) in order to implement the change request
Priority	A code that provides a recommended categorization of the urgency of the Primitive Change from Extremely important to Less important
Status	- Approved - Pending for Approval - Pending for Impact Analysis - Rejected
Impact Summary	A list of number of components that impacted as result of change request

C) "Instruction on how to implement Change Request"

Example of Change Request

Change Request Guideline	
Element	***Description***
CR-Code	CR-001
Date Reported	10-October-2015
Requested by	Purchasing Consultant (E. Willams)
Title	Improving Functionality of Creating Purchase Requisition
Submitter	J. Anderson (ERP Business Analyst)
Description	The GBI manufacturing Company requested to improve the functionality of creating purchase requisition. The functionality should be extended to provide an estimate of alternative prices of the product/service to be purchased by searching the purchase history of the company. This information enables the purchasing group to understand the approximate price of the purchase and also defines the range of price for their suppliers.
Primitive Change	Update Business Function (Creating Purchase requisition) Update Business Data (Purchase Requisition)
Priority	CR-001-1: Update Business Data (Purchase Requisition) CR-001-2: Update Business Function (Creating Purchase requisition)
Status	- Pending for Impact analysis
Impact Summary	

Please follow the steps that define in bold line in order to implement change request and analyse the impact.

Step 1) Implementation of change request

- **Select the "Change Request List" from the main menu on the top page**
- **Click on**

to create New Change Request
- **Create the change request from the example and select the change element from drop down the list**
- **Save the change request**

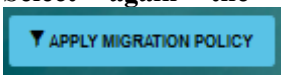
Now you are store the change request in the change request list. You can always view the detail of your change request by clicking on the edit button.

Step 2) Run Impact analysis

- **From change request list > select the change request that you already create > then Click on Impact analysis**

to define the entities that will be affected by this modification.

The page will pop up indicating the change request detail on the top page and at the bottom page; each tab represents the list of items in each category that will be affected by the modification.

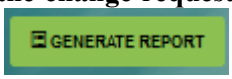
- **Close the page go back to the Change Request List**
- **Select again the change request from the list and then click on** 

to apply the migration policy for active instances and refine the Impact list.
- **"Save or Close the page"**

Step 3) View summary of Impact

To view the number of impacts for each ERP entity and compare them with the entire system you can generate a report.

Create a summary report

- **From the change request list select your change request and then click on generate report** 

- **To view summary of impact report.**

****Skip this part if you are doing any exercise**

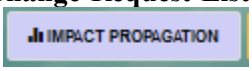
- **You can always go back and find out the details of change by clicking on**

.
- **Close the page**

- Close all the pop up pages.

Step 4) Analysis of impact propagation and Impact metrics

The information here represents the relative importance of change for each item category based on the entire system.

- **Go back "Change Request List" and select the change request from the list > then click on the** 

The information here explains the impact estimate and provides the risk level of modification (i.e. High, Medium or Low)

- **Close or Save the page**

Step 5) Propose modification strategies for each impact set

The business analyst and developer have some knowledge about the strategies for implementing change in the ERP system such as configuration strategies or code modification strategies. At this stage, the tool asks the user to propose implementation strategies for impact items.

- **To define the implementation strategies, go back to the Change Request list from the menu and select the change request > then Click on** **>**

- **Four Page will pop up.**
- **Each page asks you to determine strategies according to the number of impacts and save the propose implementation.**
- **You can enter different number according to your preference. (There is now rule at this stage that applies on proposing the right strategies as this functionality only provide you with the information between implementing different strategies. Therefore, feel free to use any strategies you prefer)**

For instance, if four processes are impacted then the user can suggest two of the processes are going to be implemented by modification of the workflow model, and two by changing the configuration setting.

Step 6) Estimate the cost modification

After defining the proposed implementation strategies, the system can calculate the estimated cost of a change request.

- **Select the change request from a list and then click on**

  The page will pop up that shows the estimate cost for each level of modification and the total cost of proposed implementation.

You can compare the cost of different change requests or different proposed strategies for one change request by using the chart at this page.

Session III. Feedback on Impact Analysis Functionality:

1) Please indicate to what extent you rate the accuracy of impact analysis tool for defining the components?

Very Low	Low	Average	High	Very High
☐	☐	☐	☐	☐

2) Do you believe that impact analysis is missing some items to capture as change impact? If yes, please indicate what?

Yes ☐ No ☐

_____

_____

_____

3) Please indicate that to what extent you agree that our approach and tool meet the following criteria:

a) The Change Impact Analysis tool identified all occurring impact items

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

b) The Change Impact Analysis tool identified all affected entity

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

c) The Change Impact Analysis provides identification of impacts set for each Item categories

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

d) The Change Impact Analysis tool identified impact sets correctly

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

4) The propose method could enhance the decision making to identify optimal implantation strategies

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

5) The Impact estimate is suitable measurement to identify the impact of change

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

6) The cost estimation is suitable measurement to compare different implementation strategies.

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

7) The proposed method could improve the customization of an ERP system more effectively

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

8) The change impact Tool analyse the impact effectively

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Session V. Implementation of change scenario

First Example of Change Scenario

Change Request Guideline	
Element	***Description***
CR-Code	CIA-0002
Date Reported	10-October-2015
Requested by	Purchasing Consultant (E. Willams)
Title	Improving Functionality of Creating Purchase Requisition
Submitter	J. Anderson (ERP Business Analyst)
Description	The GBI company has certain suppliers that they need to pay a deposit or 20% of the total price up-front before proceeding the order shipment. Therefore, during the purchasing process, two types of invoice created that one represents as a pre-payment invoice, and the other is the final invoice when the shipment and inspection of material are completed. In the standard procedure, the payment process usually started when the shipment of material is completed.
Primitive Change	Update Business Process (Purchasing Process) Update Business Data (Update Invoice Receipt for pre-payment) Update Business Function (Create Invoice receipt for vendor)
Priority	CIA-0002-1: Update Business Data (Update Invoice Receipt for pre-payment) CIA-0002-2: Update Business Function (Create Invoice receipt for vendor) CIA-0002-3: Update Business Process ((Purchasing Process)
Status	- Pending for Impact analysis
Impact Summary	

First Example: Example Change Impact Analysis Report template

Please select from an example above one type of change and implement it in the impacts analysis tool, then identify the propagation impact following by the risk level.
Please use the instruction as a manual from pervious task.

Change Request ID _____
Type of change _____
Level of Modification _____
Name of Change Item _____

Propagation of Impact set %:

Data _____%
Function _____%
Module _____%
Process _____%
Process Instance _____%
Function Instance _____%

Data Instance _____%
Total Impact Change _____ (0-1)
Level of Risk

	Low	Medium	High
Risk			

Session VI. Implementation of different Strategies for defining modification cost

Please select one of the change requests from "Change List "from the previous task and implement two different implementation strategies to calculate and compare the cost.
Please go back to report summary and filling the table below and then propose two different modification strategies

Design Impact	Data	Function	Process
Impact Number			

First suggestion: Define Cost strategies for

	Configuration	Interface-Development	Query Modification
Data			

	Configuration	Interface-Development	Code Modification	ERP Programming	Bolt-On	User exits
Function						

	Configuration	Interface-Development	Workflow Programming	ERP Programming	Bolt-On
Process					

Total Cost _____
Cost ID _____

Second suggestion: Define Cost strategies

	Configuration	Interface-Development	Query Modification
Data			

	Configuration	Interface-Development	Code Modification	ERP Programming	Bolt-On	User exits
Function						

	Configuration	Interface-Development	Workflow Programming	ERP Programming	Bolt-On
Process					

Total Cost _____
Cost ID _____

Session V. Questionnaire

Usefulness

1) The Change Impact Analysis tool helps me to identify impacts more effectively

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

2) The Change Impact Analysis tool helps me to be more productive

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

3) The Change Impact Analysis tool gives me more control over the activity when using ERP system

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

4) The Change Impact Analysis tool saves me time of identifying impact

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

5) The Change Impact analysis tool meet the requirements to identify the propagation of change in the whole system

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

6) The Change Impact Analysis tool does everything I would expect it to do

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

7) The Change Impact Analysis tool helps me understand the impact

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

8) I understand the effect of my action during customization and evolution of ERP system

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Ease of use

1) The Procedure of running Change Impact Analysis tool is easy to follow

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

2) The Change Impact Analysis tool is user friendly

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

3) I found the Change Impact analysis unnecessary complex

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

4) The tool requires the fewest step to accomplish what I want to do with it

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

5) I found the various functions in Change Impact Analysis tool were well integrated

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

6) I felt very confident using the Change Impact Analysis tool

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

7) The tool can recover from mistake quickly and easily

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

East of learning Change Impact analysis

1) I learned to use it quickly

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

2) I easy remember how to use the change impact tool

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

3) The Tool is easy to learn

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Satisfaction:

1) I am satisfied with the tool

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

2) I would recommend it to ERP Expert

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

3) It works the way that I expected

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

4) I believe this application could improve the quality of the work

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Any additional comment regarding the Tool or the procedure:

_____
_____
_____
_____

Part B) Study with Non-Expert

Session I. General Information

Please provide following information:

1. Occupation: _____ 2. Field of expertise: _____
3. Are you familiar with ERP systems? Yes ☐ No ☐
If yes, please describe your level understanding?
Excellent ☐ Very Good ☐ Good ☐ Fair ☐ Poor ☐
4. Do you have any experience using ERP systems? Yes ☐ No ☐
If yes, please provide the year of experience _____
5. Have you ever experienced using any impact analysis tool? Yes ☐ No ☐

Session II. Demonstration of the impact analysis tool

A) Explore ERP system features and functionalities of the Case study

In this session, we are providing an overview of an ERP system case study as an instantiation of the system. A case study is GBI a bike manufacturing company that implement SAP ERP in order to manage and run the activities during the process of manufacturing of Bike. GBI* Case Study demonstrates the fundamental business processes interact with SAP ERP in the area of Sales and Distribution, Material Management and Production Planning. We map the business process, functions/activities, and documents/data object from this case study into our impact analysis tool in order to identify the dependency model. This information will be used for testing purposes to define the effects of change on this particular case study.

**Global Bike Inc. has a pragmatic design philosophy that comes from its deep roots in both the off-road trail racing and long-distance road racing sports. They manufacturing bike, selling to their customer and procuring semi-finished and raw material from suppliers.*

B) Process of assessing the ERP modification

In this session, we provide you with the guidelines, before presenting the impact analysis demo. The template in the following table captures the basic requirement to proceed with the change request before implementing in ERP systems. This template is used as a guideline for assessing the changing request by the business analyst to define and prioritizing the change request.

Please take some time to read the guidelines before running any examples

C) "Instruction on how to implement Change Request"

Example of Change Request

Change Request Guideline	
Element	***Description***
CR-Code	CR-001
Date Reported	10-October-2015
Requested by	Purchasing Consultant (E. Willams)
Title	Improving Functionality of Creating Purchase Requisition
Submitter	J. Anderson (ERP Business Analyst)
Description	The GBI manufacturing Company requested to improve the functionality of creating purchase requisition. The functionality should be extended to provide an estimate of alternative prices of the product/service to be purchased by searching the purchase history of the company. This information enables the purchasing group to understand the approximate price of the purchase and also defines the range of price for their suppliers.
Primitive Change	Update Business Function (Creating Purchase requisition) Update Business Data (Purchase Requisition)
Priority	CR-001-1: Update Business Data (Purchase Requisition) CR-001-2: Update Business Function (Creating Purchase requisition)
Status	- Pending for Impact analysis
Impact Summary	

Please follow the steps that define in bold line in order to implement change request and analyse the impact.

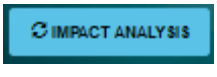
Step 1) Implementation of change request

- **Select the "Change Request List" from the main menu on the top page**
- **Click on**

to create New Change Request
- **Create the change request from the example and select the change element from drop down the list**
- **Save the change request**

Now you are store the change request in the change request list. You can always view the detail of your change request by clicking on the edit button.

Step 2) Run Impact analysis

- **From change request list > select the change request that you already create > then Click on Impact analysis** 

to define the entities that will be affected by this modification.

The page will pop up indicating the change request detail on the top page and at the bottom page; each tab represents the list of items in each category that will be affected by the modification.

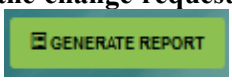
- **Close the page go back to the Change Request List**
- **Select again the change request from the list and then click on**

to apply the migration policy for active instances and refine the Impact list.
- **"Save or Close the page"**

Step 3) View summary of Impact

To view the number of impacts for each ERP entity and compare them with the entire system you can generate a report.

Create a summary report

- **From the change request list select your change request and then click on generate report** 

- **To view summary of impact report.**

****Skip this part if you are doing any exercise**

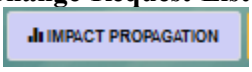
- **You can always go back and find out the details of change by clicking on**

.
- **Close the page**

- Close all the pop up pages.

Step 4) Analysis of impact propagation and Impact metrics

The information here represents the relative importance of change for each item category based on the entire system.

- **Go back "Change Request List" and select the change request from the list > then click on the** 

The information here explains the impact estimate and provides the risk level of modification (i.e. High, Medium or Low)

- **Close or Save the page**

Step 5) Propose modification strategies for each impact set

The business analyst and developer have some knowledge about the strategies for implementing change in the ERP system such as configuration strategies or code modification strategies. At this stage, the tool asks the user to propose implementation strategies for impact items.

- **To define the implementation strategies, go back to the Change Request list from the menu and select the change request > then Click on**

>

- **Four Page will pop up.**
- **Each page asks you to determine strategies according to the number of impacts and save the propose implementation.**
- **You can enter different number according to your preference. (There is now rule at this stage that applies on proposing the right strategies as this functionality only provide you with the information between implementing different strategies. Therefore, feel free to use any strategies you prefer)**

For instance, if four processes are impacted then the user can suggest two of the processes are going to be implemented by modification of the workflow model, and two by changing the configuration setting.

Step 6) Estimate the cost modification

After defining the proposed implementation strategies, the system can calculate the estimated cost of a change request.

- **Select the change request from a list and then click on**

  The page will pop up that shows the estimate cost for each level of modification and the total cost of proposed implementation.

You can compare the cost of different change requests or different proposed strategies for one change request by using the chart at this page.

Session III. Implementation of change scenario

First Example of Change Scenario

Change Request Guideline	
Element	***Description***
CR-Code	CIA-0002
Date Reported	10-October-2015
Requested by	Purchasing Consultant (E. Willams)
Title	Improving Functionality of Creating Purchase Requisition
Submitter	J. Anderson (ERP Business Analyst)
Description	The GBI company has certain suppliers that they need to pay a deposit or 20% of the total price up-front before proceeding the order shipment. Therefore, during the purchasing process, two types of invoice created that one represents as a pre-payment invoice, and the other is the final invoice when the shipment and inspection of material are completed. In the standard procedure, the payment process usually started when the shipment of material is completed.
Primitive Change	Update Business Process (Purchasing Process) Update Business Data (Update Invoice Receipt for pre-payment) Update Business Function (Create Invoice receipt for vendor)
Priority	CIA-0002-1: Update Business Data (Update Invoice Receipt for pre-payment) CIA-0002-2: Update Business Function (Create Invoice receipt for vendor) CIA-0002-3: Update Business Process ((Purchasing Process)
Status	- Pending for Impact analysis
Impact Summary	

First Example: Example Change Impact Analysis Report template

Please select from an example above one type of change and implement it in the impacts analysis tool, then identify the propagation impact following by the risk level.
Please use the instruction as a manual from pervious task.

Change Request ID _____
Type of change _____
Level of Modification _____
Name of Change Item _____

Propagation of Impact set %:

Data _____%
Function _____%
Module _____%
Process _____%
Process Instance _____%
Function Instance _____%

Data Instance _____%
Total Impact Change _____ (0-1)
Level of Risk

	Low	Medium	High
Risk			

Session IV. Implementation of different Strategies for defining modification cost

Please select one of the change requests from "Change List "from the previous task and implement two different implementation strategies to calculate and compare the cost.
Please go back to report summary and filling the table below and then propose two different modification strategies

Design Impact	Data	Function	Process
Impact Number			

First suggestion: Define Cost strategies for

	Configuration	Interface-Development	Query Modification
Data			

	Configuration	Interface-Development	Code Modification	ERP Programming	Bolt-On	User exits
Function						

	Configuration	Interface-Development	Workflow Programming	ERP Programming	Bolt-On
Process					

Total Cost _____
Cost ID _____

Second suggestion: Define Cost strategies

	Configuration	Interface-Development	Query Modification
Data			

	Configuration	Interface-Development	Code Modification	ERP Programming	Bolt-On	User exits
Function						

	Configuration	Interface-Development	Workflow Programming	ERP Programming	Bolt-On
Process					

Total Cost _____
Cost ID _____

Session V. Questionnaire

Usefulness

9) The Change Impact Analysis tool helps me to identify impacts more effectively

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

10) The Change Impact Analysis tool helps me to be more productive

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

11) The Change Impact Analysis tool gives me more control over the activity when using ERP system

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

12) The Change Impact Analysis tool saves me time of identifying impact

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

13) The Change Impact analysis tool meet the requirements to identify the propagation of change in the whole system

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

14) The Change Impact Analysis tool does everything I would expect it to do

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

15) The Change Impact Analysis tool helps me understand the impact

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

16) I understand the effect of my action during customization and evolution of ERP system

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Ease of use

8) The Procedure of running Change Impact Analysis tool is easy to follow

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

9) The Change Impact Analysis tool is user friendly

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

10) I found the Change Impact analysis unnecessary complex

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

11) The tool requires the fewest step to accomplish what I want to do with it

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

12) I found the various functions in Change Impact Analysis tool were well integrated

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

13) I felt very confident using the Change Impact Analysis tool

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

14) The tool can recover from mistake quickly and easily

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

East of learning Change Impact analysis

4) I learned to use it quickly

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

5) I easy remember how to use the change impact tool

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

6) The Tool is easy to learn

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Satisfaction:

5) I am satisfied with the tool

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

6) I would recommend it to ERP Expert

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

7) It works the way that I expected

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

8) I believe this application could improve the quality of the work

Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Any additional comment regarding the Tool or the procedure:

_____
_____
_____
_____

APPENDIX. D IMPLEMENTATION MICROFLOW, DOMAIN

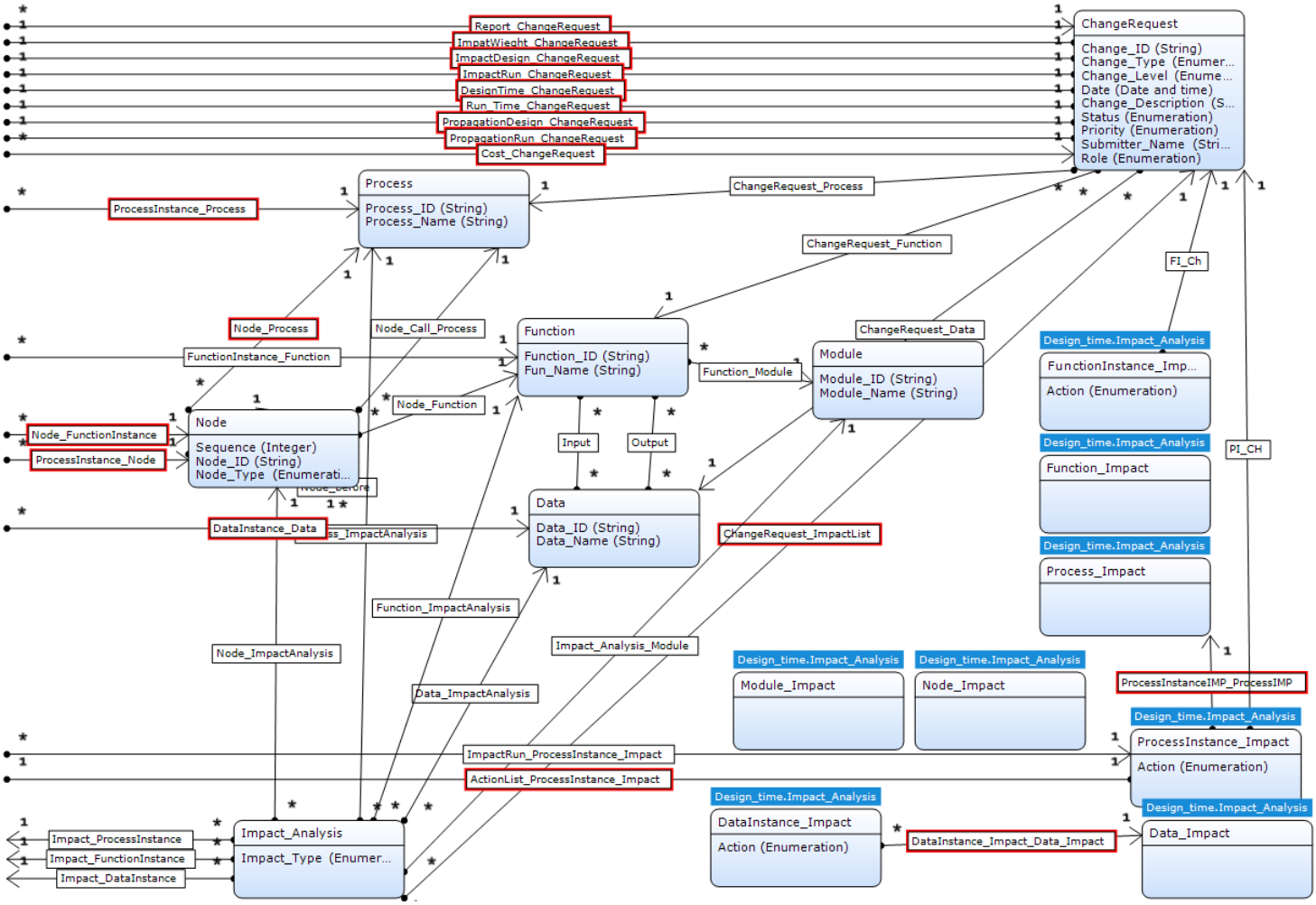


Figure A-D 1 Design-Time Domain Model & Change Impact

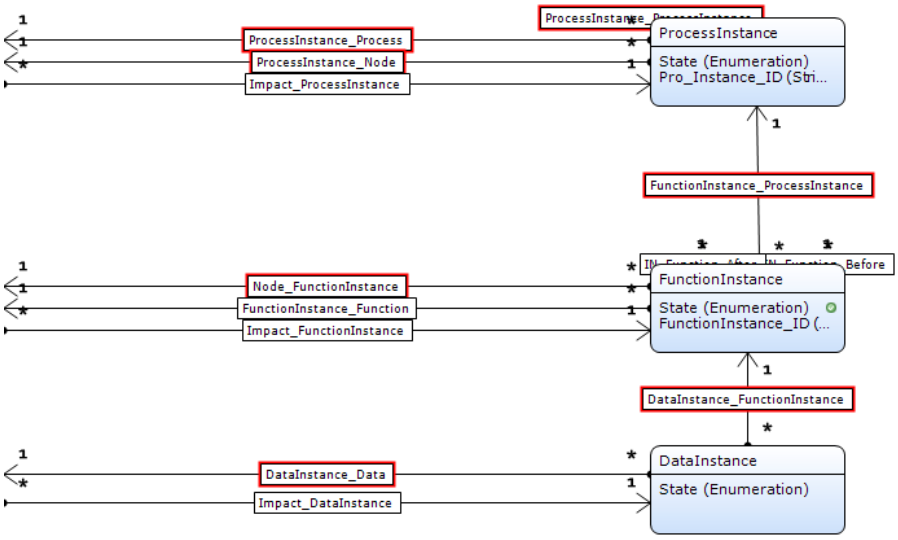


Figure A-D 2 Run-Time Domain Model

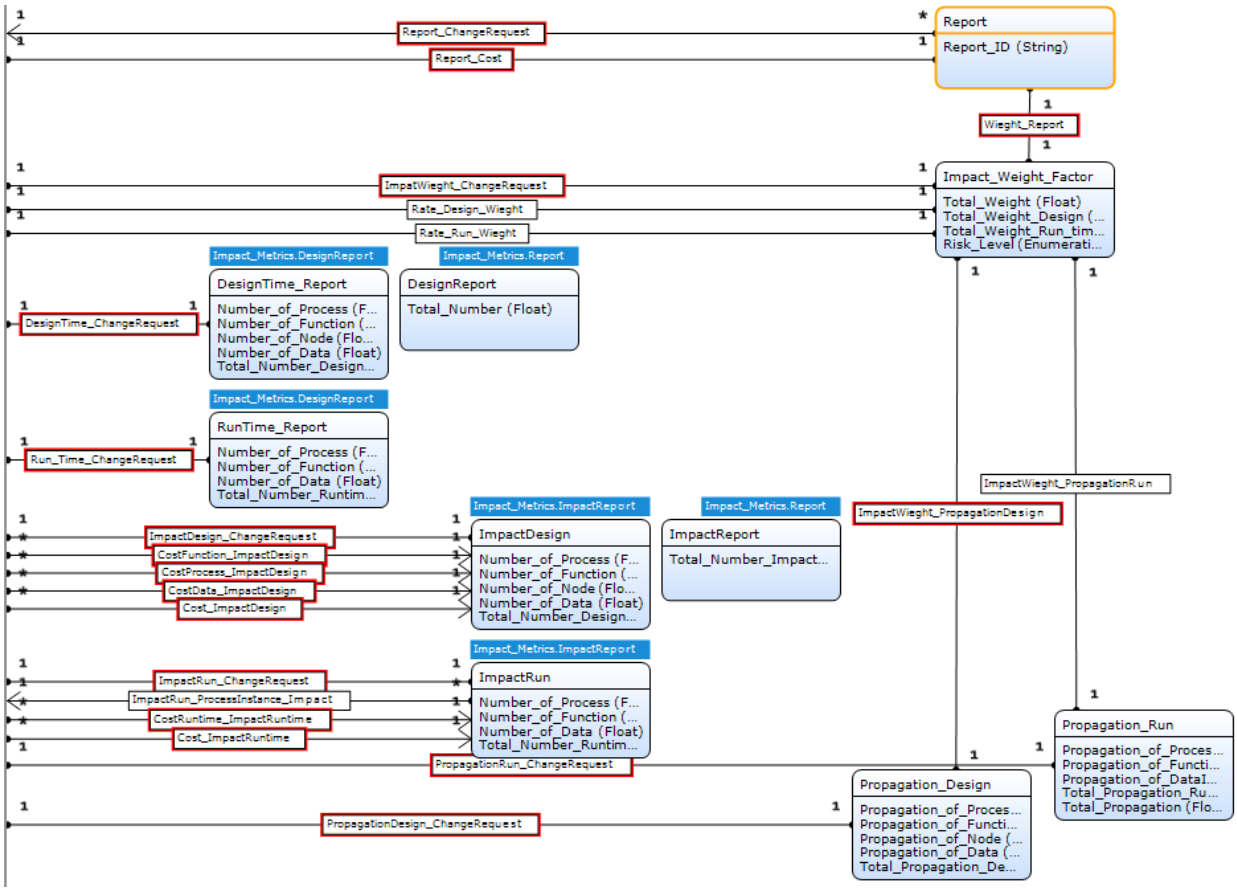


Figure A-D 3 Impact Report Domain Model

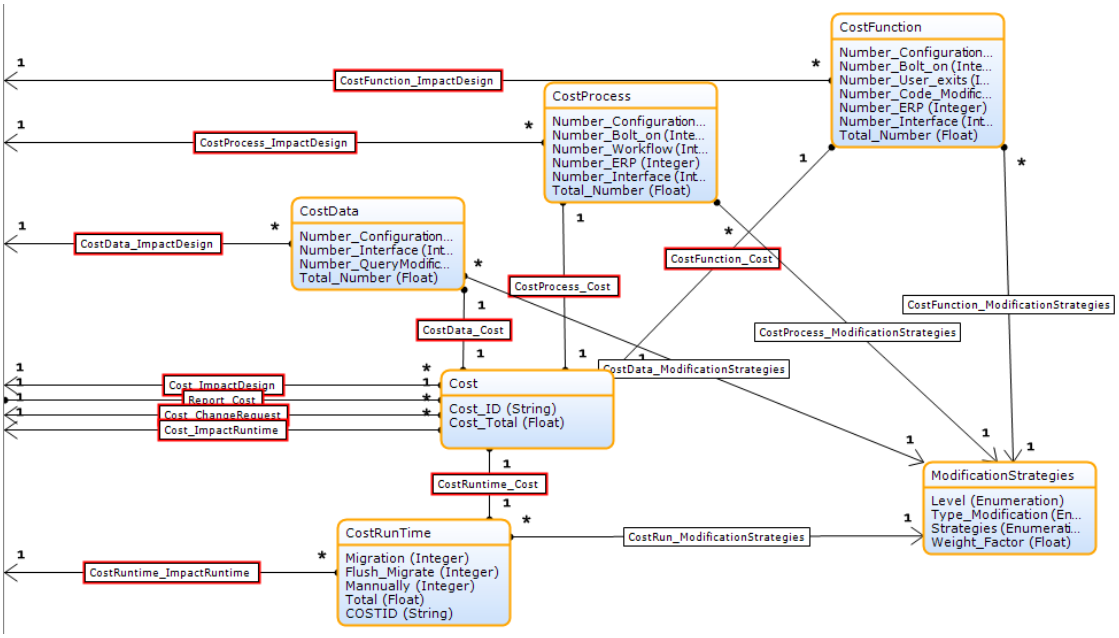


Figure A-D 4 Cost Calculation Domain model

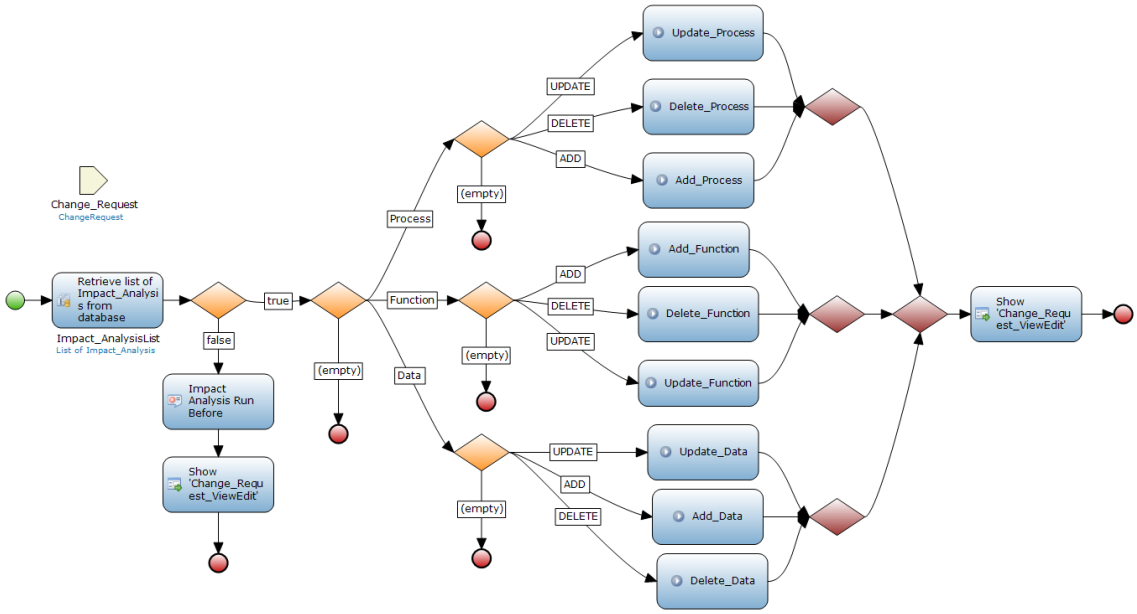


Figure A-D 5 Impact analysis mechanisms

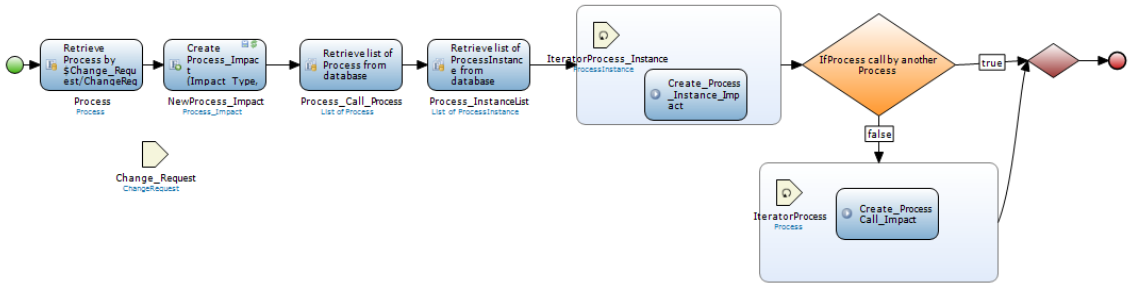


Figure A-D 6 Update Business Process (Algorithm 5)

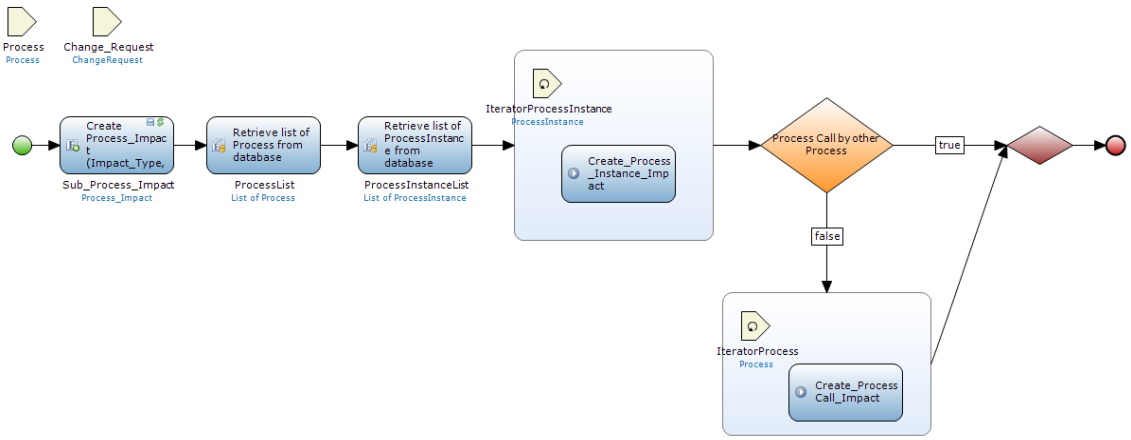


Figure A-D 7 Microflow for (Process Call) Algorithm 3

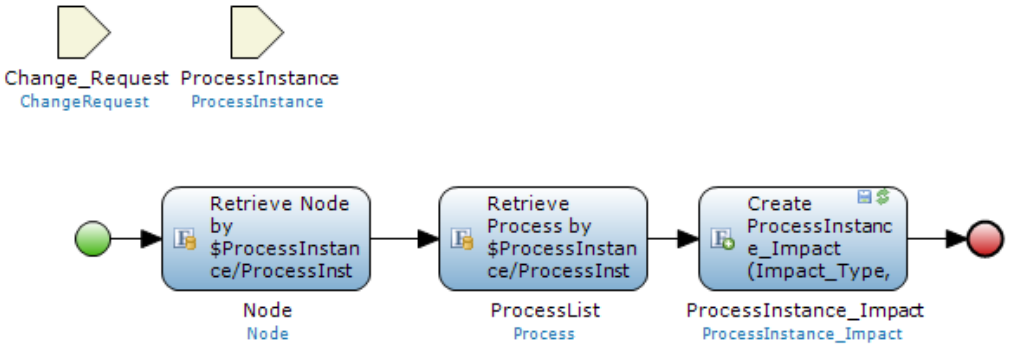


Figure A-D 8 Microflow for Create Process Instance (Algorithm 3 & 5)

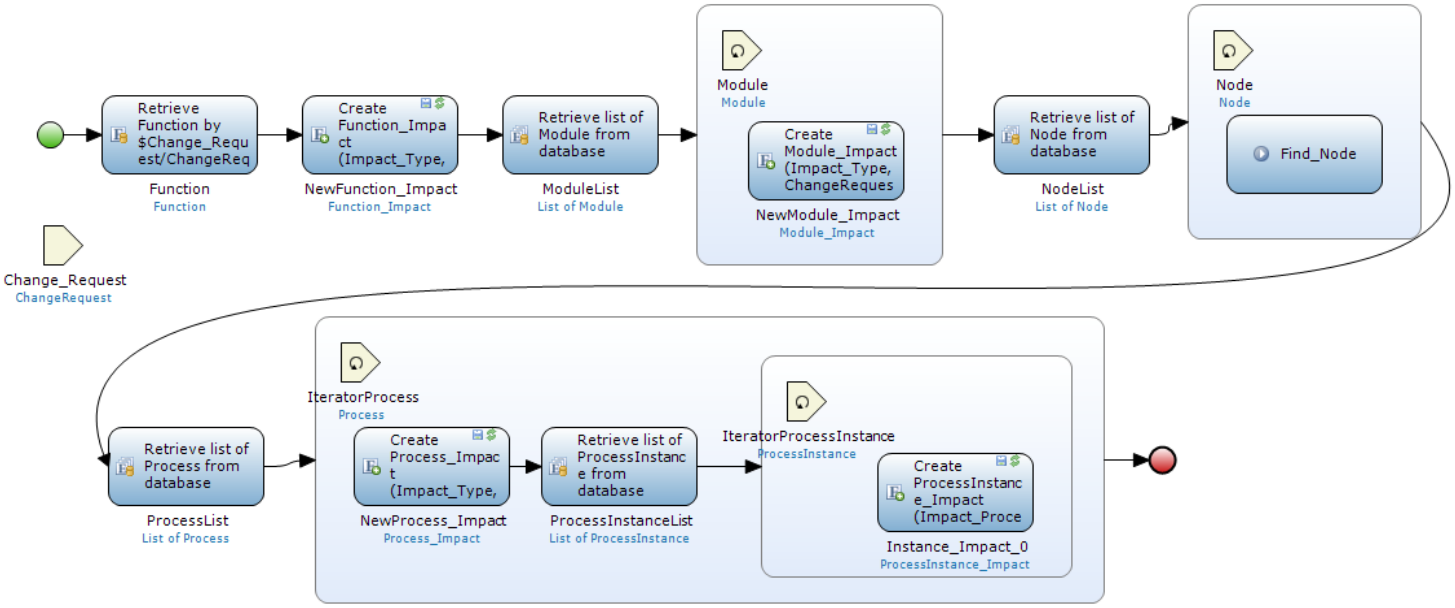


Figure A-D 9 Update Business Function (Algorithm 4)

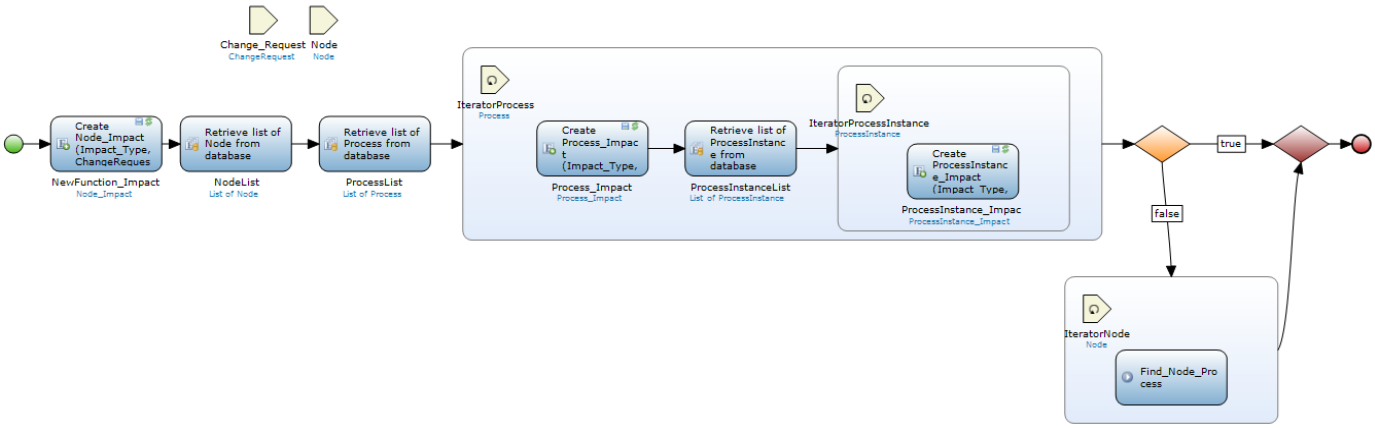


Figure A-D 10 Microflow Find Node (Algorithm 4)

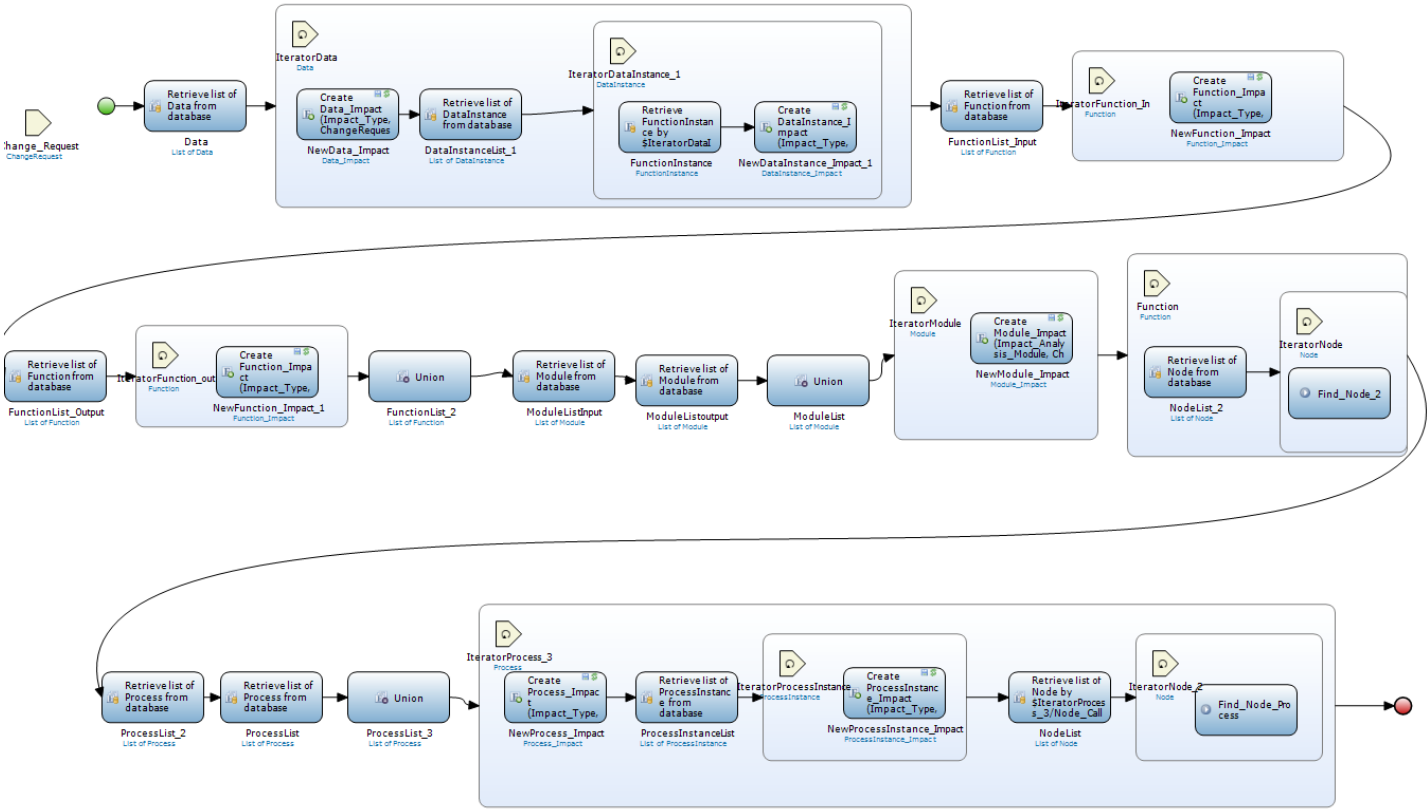


Figure A-D 5 Update Business Data (Algorithm 2)

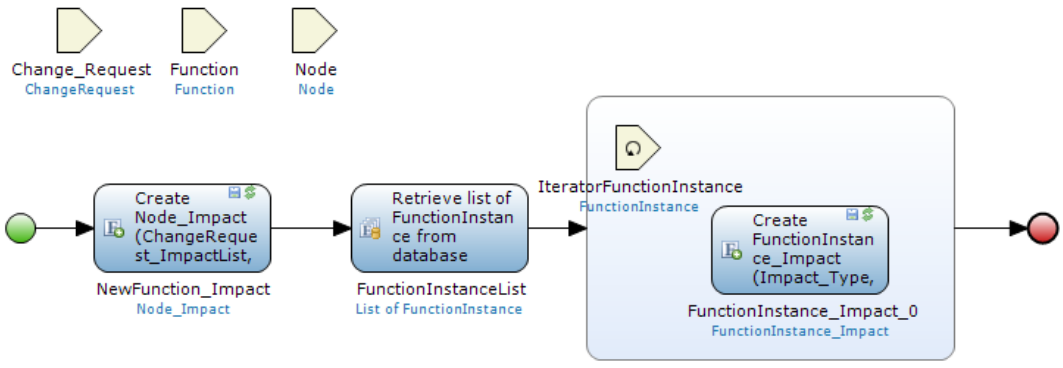


Figure A-D Microflow Find Node-2 (Algorithm 2

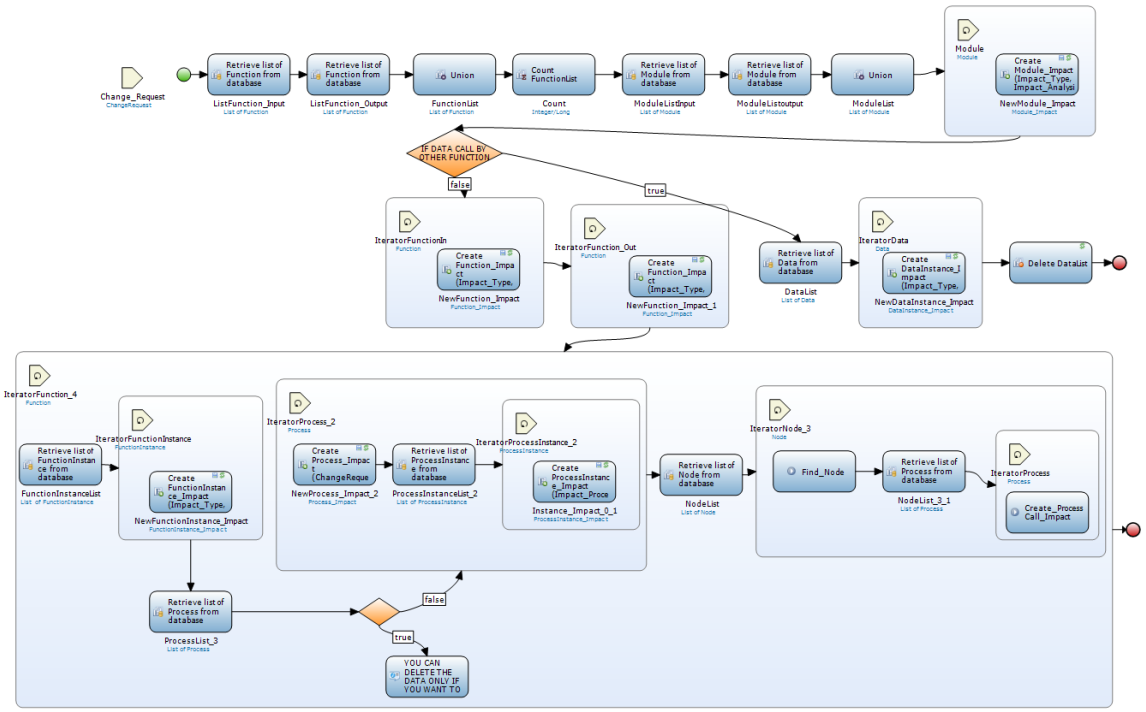


Figure A-D 5 Update Business Data (Algorithm 6)

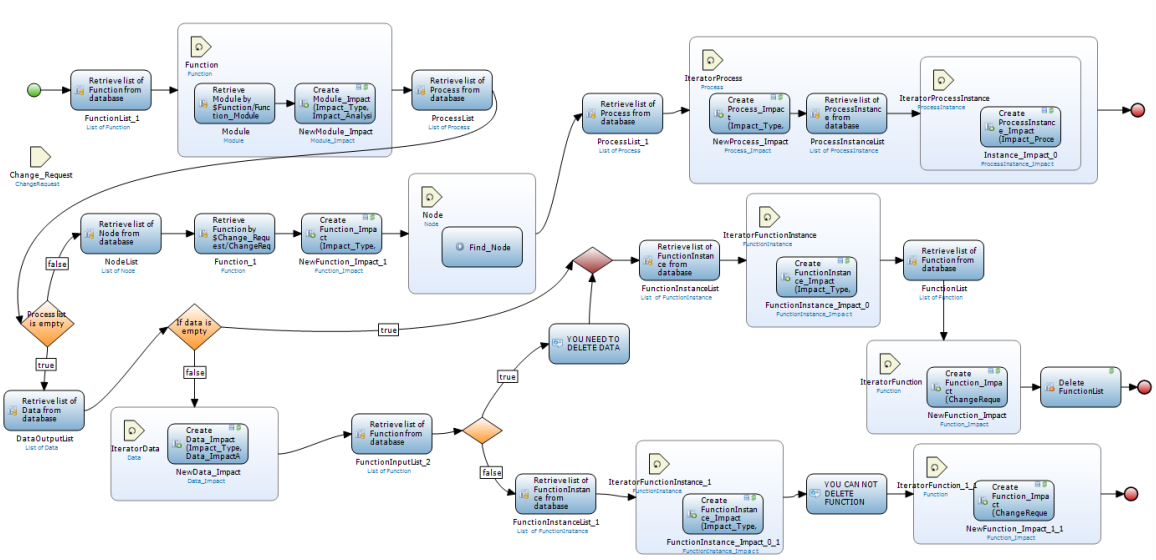


Figure A-D 5 Update Business Function (Algorithm 7)

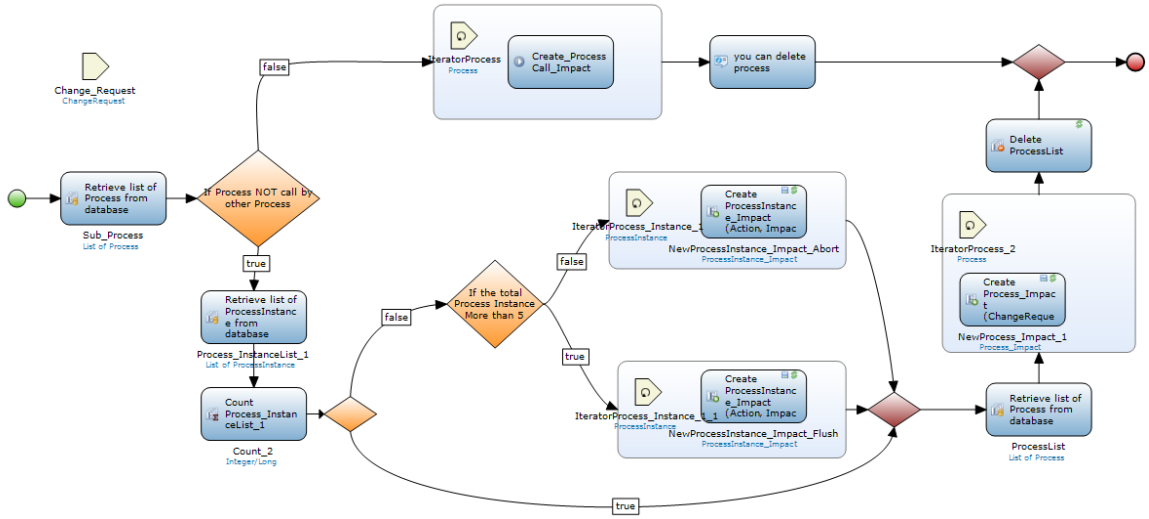


Figure A-D 5 Update Business Process (Algorithm 8)

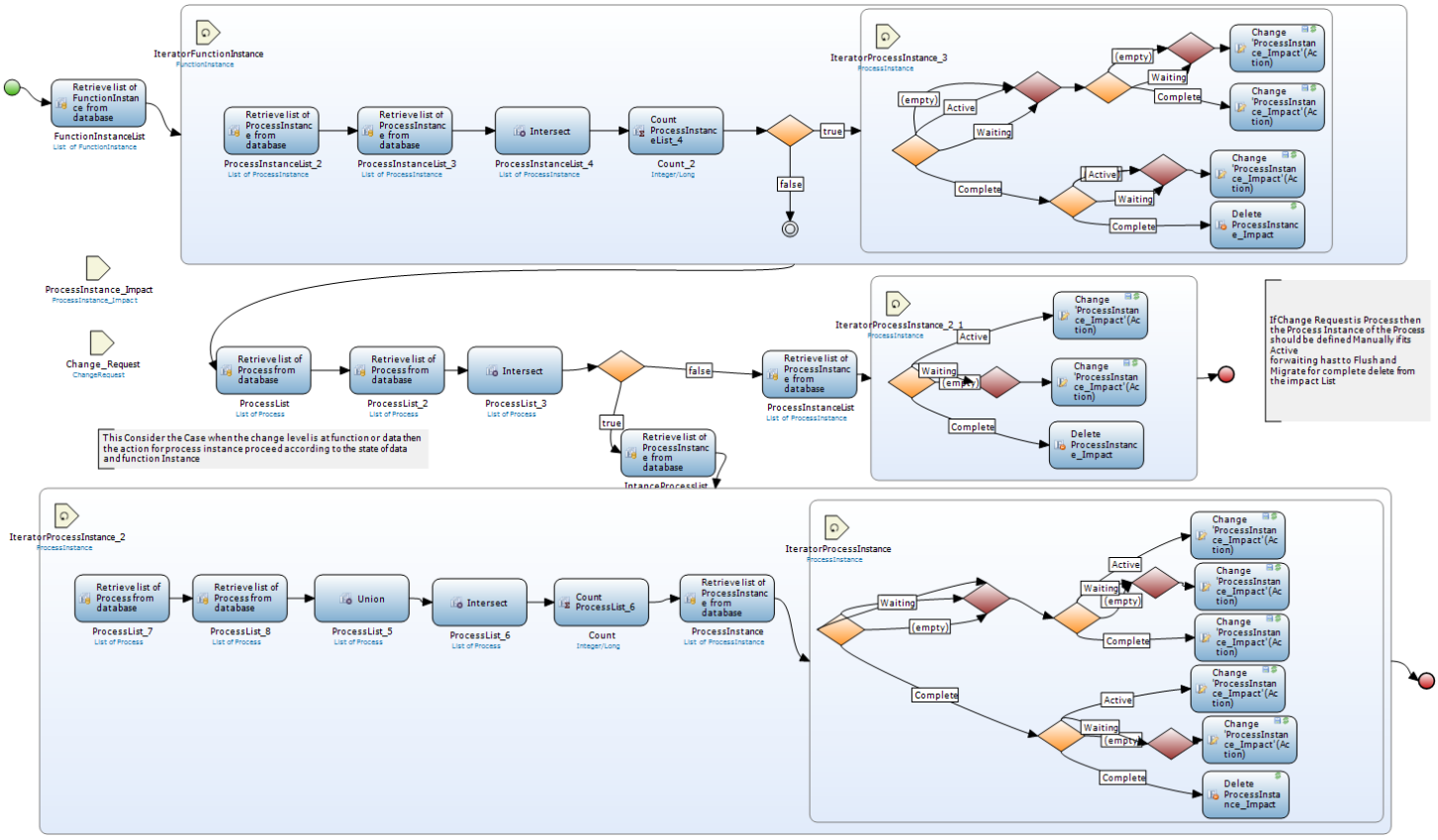


Figure A-D 5 Apply Migration Policies for Process Instance

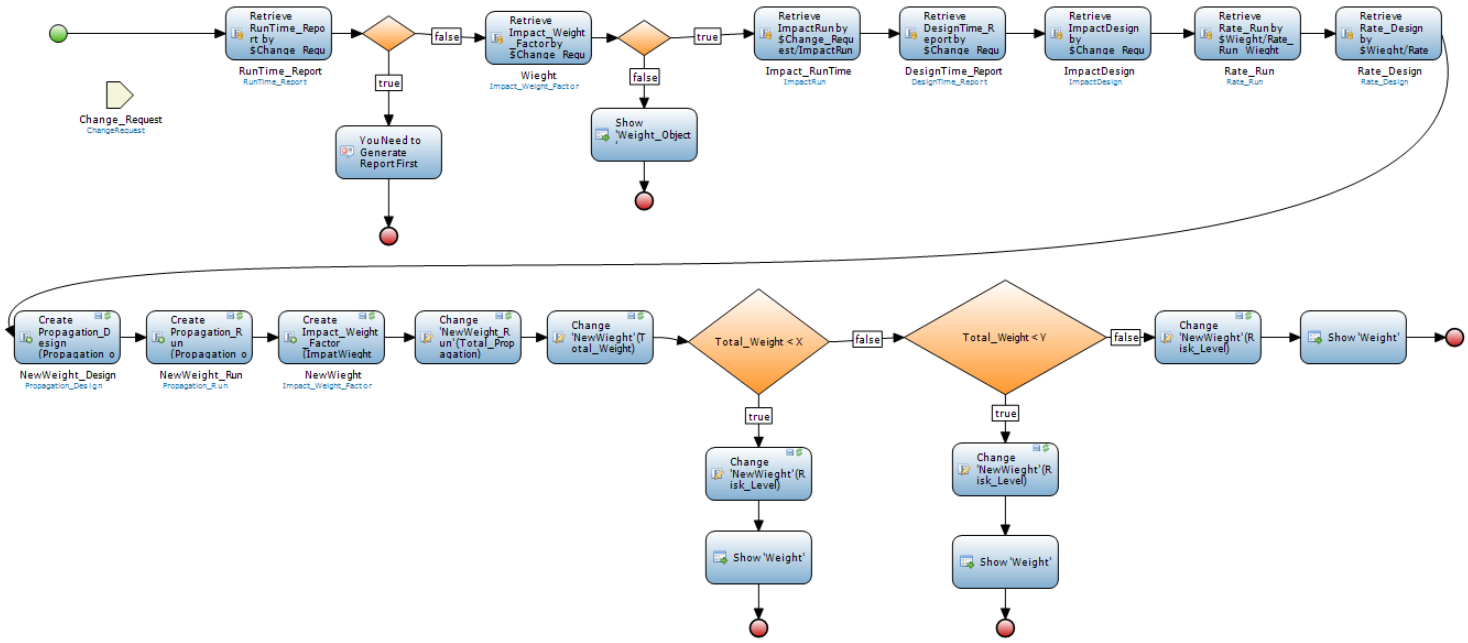


Figure A-D 5 Calculate the Total Impact

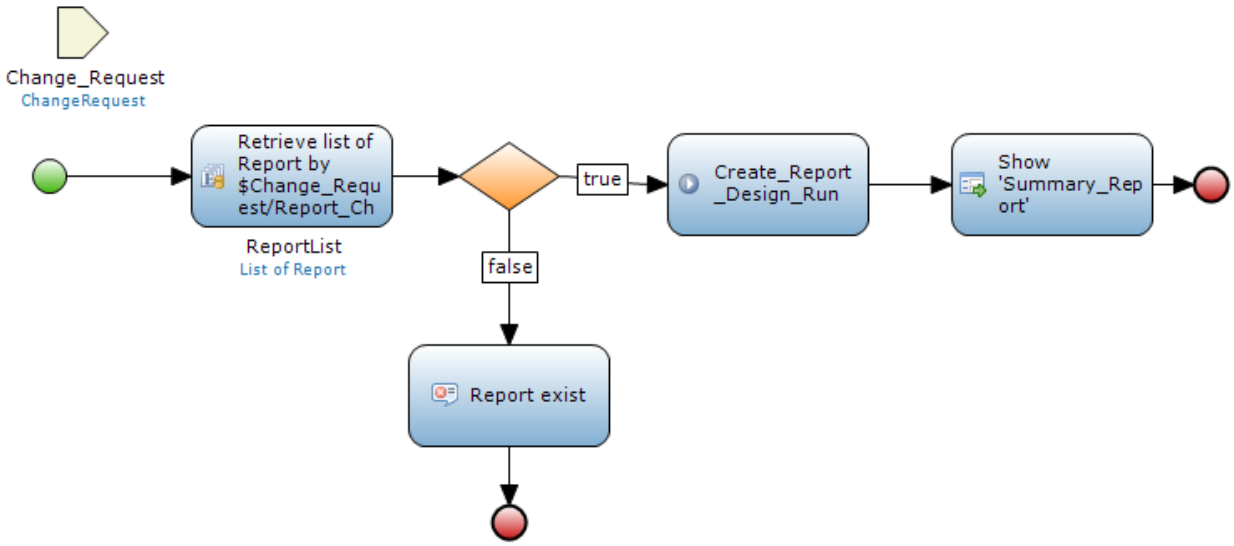


Figure A-D 5 View Impact Report

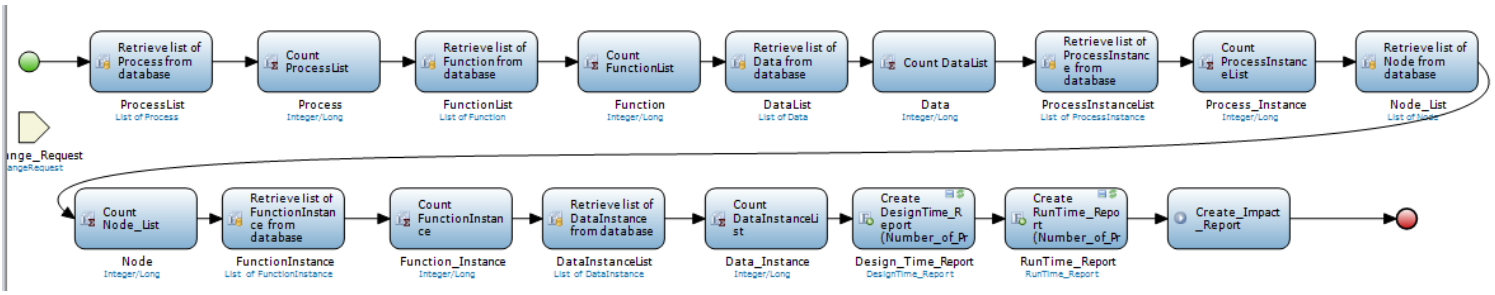


Figure A-D 5 Create Impact Report for Run Time Item

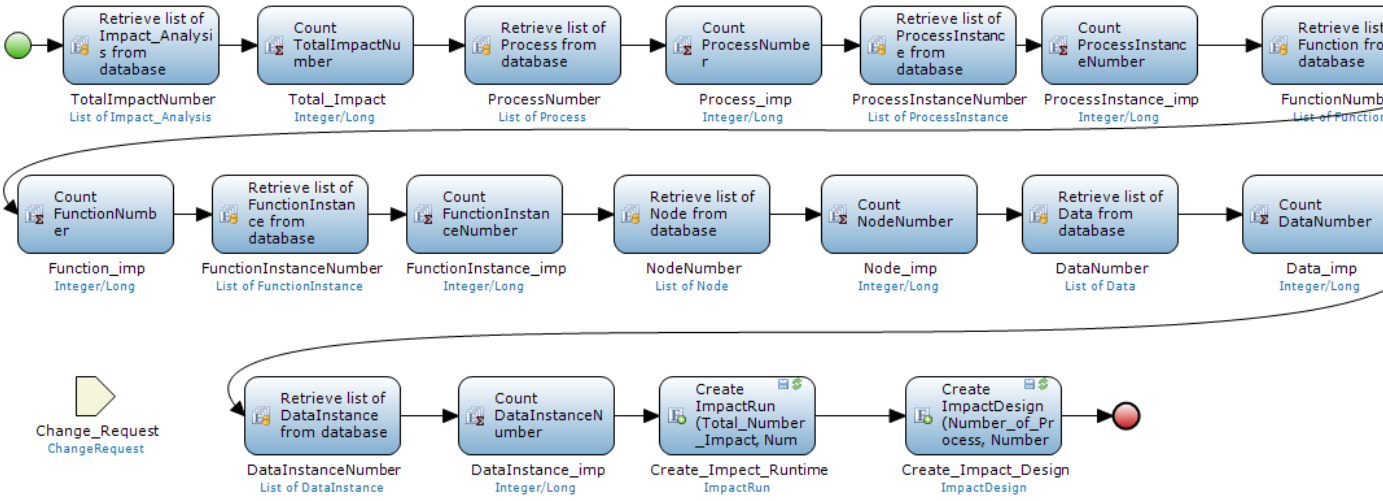


Figure A-D 5 Create Impact Report for Design Time Item

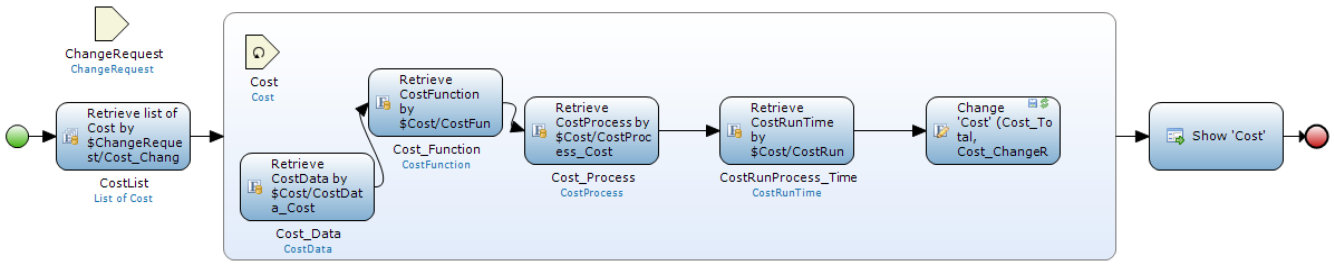


Figure A-D 5 Calculate the total Cost

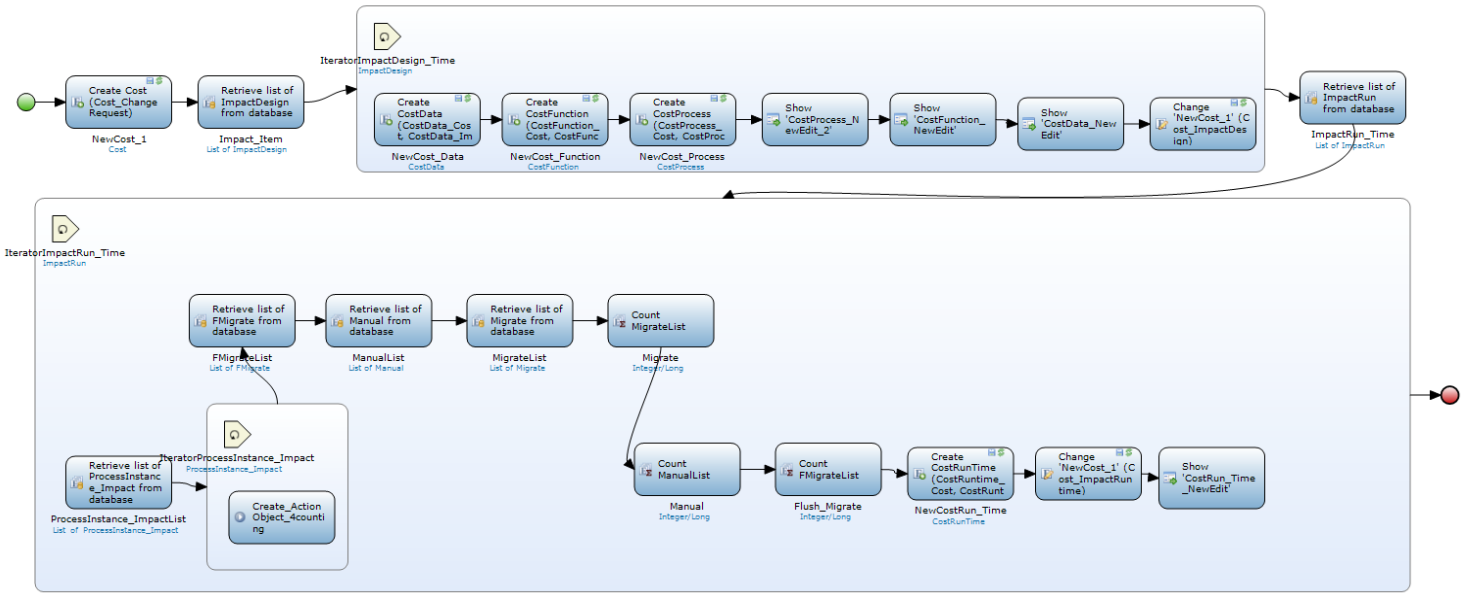
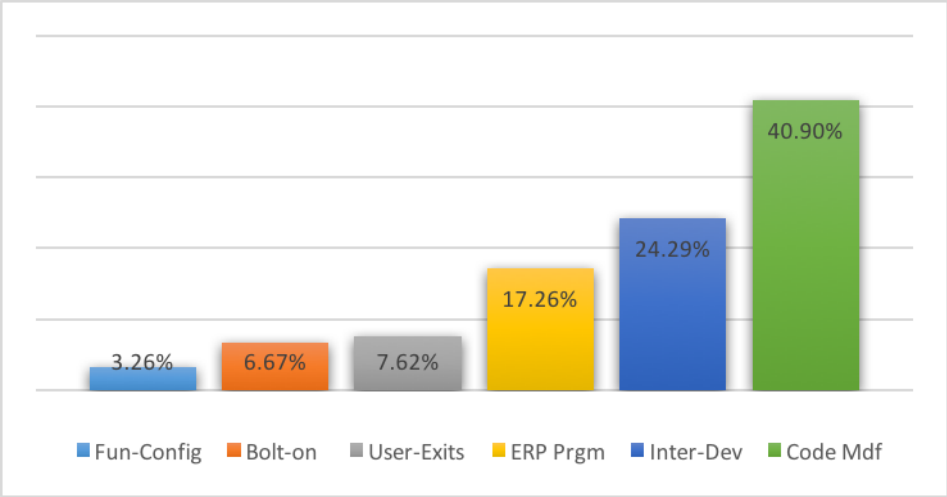


Figure A-D 5 Calculate the Assign Implementation

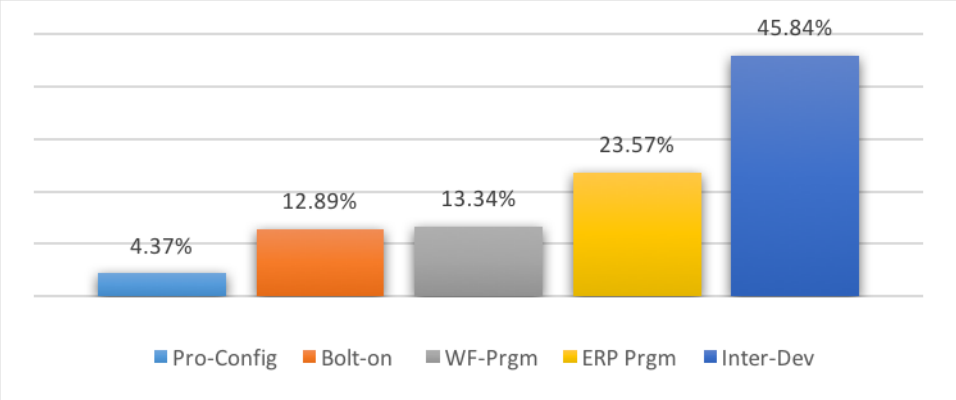
APPENDIX. E RESULT OF THE STUDY

Result from AHP- Method

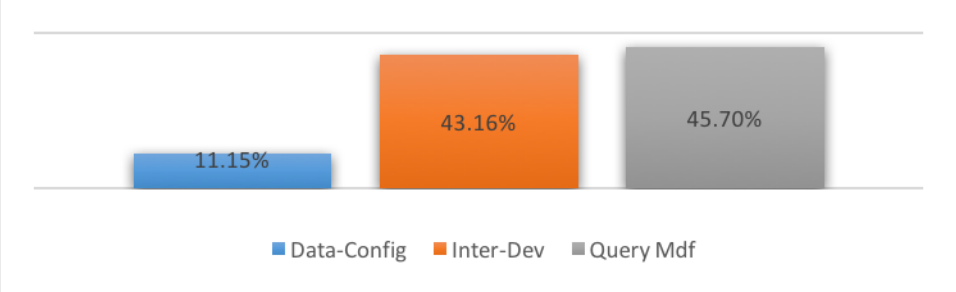
Business Function Level



Business Process Level



Business Data Level



Interview transcript with ERP professional

Table 9-1 Interviewee 1

Interviewee Background	*Position*: Head of training ERP at GSK *Field of Expertise*: Education and training (Netweaver) *Experience in ERP system*: More than 10 Years
Q1) Any experience using impact analysis tool in your ERP system	We use Solution Manager but not for assessing the impact. We only use it for to documentation repository to process our documentation and not using for process mapping and linking the processes to define the dependencies such as roles and functionality. We mainly used it for document management and process definition, which is used for specifying our process and to attach what business blueprints looks like for that particular business process, such as standard operating process solution (SOPs). Also, it's not very easy to use this application as it is even more complicated to understand and use than SAP application itself.
Q2) Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	We got extensive business process especially in the field of pharmacist industry. We document our solution in a standard operating procedure where we have hundred standard procedures. So they are detailed step-by-step level (i.e. in terms of activities) and role level. As well as that we have set of business process solution document BPSD, which is a high-level documentation, and we contain Microsoft Visio within there.
Q3) Assessment through standard set of steps/activities or a process to trace the impact of modification	When we have a change request solution, it will only evaluate if it is driven by the business or SAP program. Such as we are operating in lots of countries, for instance, we receive a change request from Italy that needs customization of the SAP system due to the change in legal regulation of the country. So they need to come to main office (i.e. which us) and through our process team to assess the modification. At that point we analyse the impact of this particular request. In term of what impact assessment look likes, by first check if the modification is in any of the categories of legal, fiscal, or regulatory. Then we assess the impact by imply into the templates that include three-page documentation which needs to go around to the various teams to evaluate the modification deeply including the implementation, cost and training
Q4) Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	For mapping we use business process documentation (i.e. BPSD and Visio) and for the change history we use change control documentation (i.e. Template) and some traceability metric
Q5) Importance of having structured approach and Tool for assessing the impact	It is critical particularly for the businesses like a pharmacist. The main reason is that when you become dependent upon SAP whenever you want one small change to the system, it may affect another part of the system and propagate extensively due well-integrated business processes and data. So it is important for us if you cannot assess the impact correctly so the tool as the solution can increase the accuracy of impact analysis effectively.

Table 9-2 Interviewee 2

Interviewee Background	*Position:* Operational Excellence lead EMEA (Macquarie Bank) *Field of Expertise*: Change management and Six sigma certified *Experience in ERP* system: More than 15 Years
Q1) Any experience using impact analysis tool in your ERP system	Not anything particular for assessing the modification in ERP system.
Q2) Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	We use Microsoft Visio for mapping the (as-is) business processes and using the long sheet (known as brown paper along the room) to discuss the changes in detail and then defining all the interaction and dependencies of that particular business process in a very high-level form. Then after discussion we map the new business processes (to-be) and compare them together in the hard copy. The main reason for taking this approach is to highlight all the issues in the business processes informally before convert them in the Visio. Once all the parts and dependencies are defined and then transfer the mapping for both processes from brown papers to the Visio file. Therefore, our strategy in determining change and their impacts is more based on expert judgment rather than any formal way.
Q3) Assessment through standard set of steps/activities or a process to trace the impact of modification	We implied the six-sigma methodology to define and locate where the problems are in the system, and then gathering people such as developer analyst to assess if they can fit this modification with the system for this purpose or not. If we need to apply the ERP customization, then it is important to define what need to change in the system.
Q4) Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	We use version control documentation (i.e., change template) and list of business process retrieving from Microsoft Visio
Q5) Importance of having structured approach and Tool for assessing the impact	Our approach is to improve the business process. We want the user to come to us and look at the process and inform us what can we do to improve the business processes. So this means that we need to facilitate and gather the team to analyse that. Therefore, it is important for us to be able to assess the change more effectively through automated solution and standard approach.

Table 9-3 Interviewee 3

Interviewee Background	*Position*: Finance Director at IRIS Software Group *Field of Expertise:* Change management (and implementation of the SAP, JD Edward, MD AG, CRM) *Experience in ERP system:* More than 15 Years
Any experience using impact analysis tool in your ERP system	Do not use any Impact analysis tool
Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	As a business analyst, I have experience in implementation of ERP at both public sector and multi-international sector where the organisation is a part of thousand other companies in the group. Therefore, we usually faced with the situation where there the organisations have some requirement that needs some level of modification in ERP system. So the way that we deal with these situations is to use process mapping tool like Visio or RPIW (i.e. rapid process improvement workshop) and gather various employee from different sector to analyse the problem. During the session, the ERP implementation consultant has to define where the problems are in the business process and highlight them accordingly. During the mapping of processes, we also used the business process re-engineering techniques for design the business process in Visio.
Assessment through standard set of steps/activities or a process to trace the impact of modification	We have a standard procedure during the change process. All the sets such as process mapping for the as-is and to-be must be signed and authorized to proceed further. Then we have detailed documentation about the transition and planning on how to apply the change that again this need to be approved and agreed. The transition document indicates what needs to change in the process. This is more based on the expert judgment and experience rather than any automated mechanism. One example of the change process is back to the two years ago where we were in a situation on improving the financial reporting process that requires us to change the whole procedure. At that time, 12 people sat down and discussed the problem for about five-day and then mapping the whole thing, indicating where the problems are and considered the solution to apply for new requirements. Then after six months' time, we were in a situation to migrate the system to the new business process.
Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	We first mapped the process as -is then we say what the actual requirement needs to be done. Therefore, there is the structured procedure to it and standard documentation to use each time we end up with change in ERP system. It is important for us to documenting the whole procedures as-is and to-be of process mapping in order to reuse it during the modification process.
Importance of having structured approach and Tool for assessing the impact	Depending on the solution we believe that impact analysis tool can be effective and efficient solution as long as the organisation can run the tool without asking from an external consultant to run it for you.

Table 9-4 Interviewee 4

Interviewee Background	*Position:* Senior business analysis City University *Field of Expertise*: Change management, ERP, Business Intelligent *Experience in ERP system:* 25 Years
Q1) Any experience using impact analysis tool in your ERP system	Do not use any Impact analysis tool
Q2) Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	In the previous project, we did as-is and to-be process mapping in order to identify where the issues are in the process. We used Microsoft Visio to document the process mapping of the current system and the interactions with peoples and other systems. We also apply the same technique for the new system and then compare both documentation, and identify the significant changes. Thus it is more based on the business analyst knowledge and experience rather than a formal mechanisms or tool
Q3) Assessment through standard set of steps/activities or a process to trace the impact of modification	After mapping the business processes of two types, we outsourced change management consultant from another company as a part of our team. Once the significant changes identified change management team, have to ensure that everyone can understand the change and what new system is going to do. Then this goes to the authorization committees to make the decision to evaluate if it's beneficial by applying the change in the system or change the organisational procedures. However, there are lots of emphases to not make any changes in the ERP system due to the implementation cost and maintaining the change during the system upgrades.
Q4) Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	We identified where the changes are and measure the time for the execution and compared the both process. The main source of information for us is the process maps to analysis the change in Visio.
Q5) Importance of having structured approach and Tool for assessing the impact	That would be very useful which help to take the personal judgment out of it and the impact of modification would be assessed in more mechanical, logical and consistent approach. This way can help in most of the time where the business analyst missing some part and the interaction, therefore, having a tool could enhance the measurement more accurately and correctly.

Table 9-5 Interviewee 5

Interviewee Background	*Position*: Management Consultant (Helping to align Business and enterprise IT) *Field of Expertise*: Change management, ERP system implementation *Experience in ERP system*: 20 Years
Q1) Any experience using impact analysis tool in your ERP system	Not any at the moment, we only used impact analysis for change management which is more reflects on the people and organisation rather than change on ERP system.
Q2) Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	We identify the requirements and mapping them to the business process. We have application consultant that works within a team and then they will identify whether it is a standard configuration or some level of development. If we recognize that modification involves development effort, then we typically look at some business justification such that why it needs to apply this change, why not a standard configuration and what's the benefits in term of cost of development and maintenance? During the process we use process-modelling tool like Visio, or Bizagi (open source equivalent), or more sophisticated modelling like ARIS. It is important to indicate that SAP has the reaches set of tool and partners compare to some of the other ERP products.
Q3) Assessment through standard set of steps/activities or a process to trace the impact of modification	First it is essential to understand the requirements then you need to challenge that it is good in practice or not, our approach is constructed more based on expert judgment through the comparison of different model and experience. We have some procedure but not in very formal and constructive standard approach. It is about mapping the process when the process is eliciting the requirements, identify how these requirements can be mapped with the solution and then deciding on whether it is standard configuration or development for planning the modification and assessing what sort of the solutions are suitable for that.
Q4) Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	So we don't use any dependency model, and it is more based on people experience and judgments. We certainly capture as a non-standard requirement, which in most cases are subjective. Such a financial group wants to do something in a particular way, and then you could only push back so many times. After determining the requirements for the business process, then passed this information and discussed with the application consultancy in a form of workshop environment. They evaluate the conditions where the configuration is not feasible, and they might need some customization. Based on the assessment they look at the business benefit and implementation cost of ERP customization. So over all its subjective evaluation of each change request.
Q5) Importance of having structured approach and Tool for assessing the impact	It is important to have impact analysis potentially it is useful, to have a tool and structure way depending on the implementation and type of assessment. However, this comes to the point where there is a need for a consultant to define and assess the change. But in particular, it is much helpful to have a tool and approach that developed and evaluated through the scientific approach.

Table 9-6 Interviewee 6

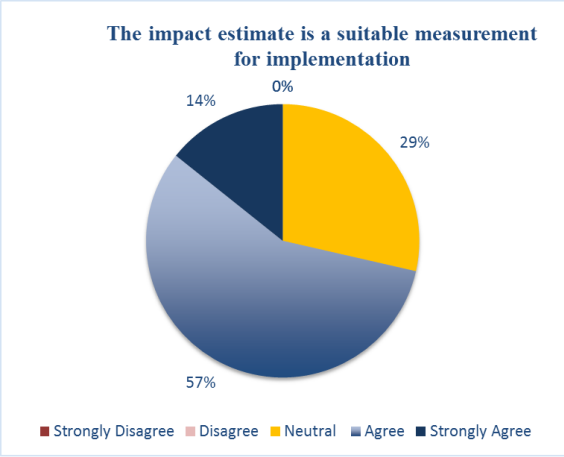
Interviewee Background	*Position*: Analyst Developer *Field of Expertise:* ERP Operational process and system tester *Experience in ERP system*: 7 Years
Q1) Any experience using impact analysis tool in your ERP system	Not any at the moment.
Q2) Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	We are not using any major tool for that, we use Visio for documentation, but more everything is based on our judgment during the change process rather than using any particular tool for that. We only use function specification for process mapping, which is used as the perception of what system has. As an analyst what we majorly do is if there is the problem, then we evaluate it according to the input and output compare to the function specification documents. If we noticed any problem or inconsistency in the system, then we report that to system developer and support team to analyse it further and assess what other part of the system is going to be affected.
Q3) Assessment through standard set of steps/activities or a process to trace the impact of modification	If the system addresses any errors or undesired output, so we first go through the process map. Once we get an understanding of the as-is process changes, then we go through the functional specification documents that provide us with all the interaction to the database. This documents gives us an understanding of the dependencies with other objects (i.e., input and output). Then we make any assumption about the gaps between the new requirements and as-is process. So my job is to analyse the change in more high-level and translates them for the ERP developer to investigation further at the low-level analysis (i.e. such as change at source code).
Q4) Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	We have functional specification document and process documentation that we used during the change process.
Q5) Importance of having structured approach and Tool for assessing the impact	For us impact analysis an important task, to have automated solution can help us to identify the dependencies faster without going through 700 documents to specify what would change.

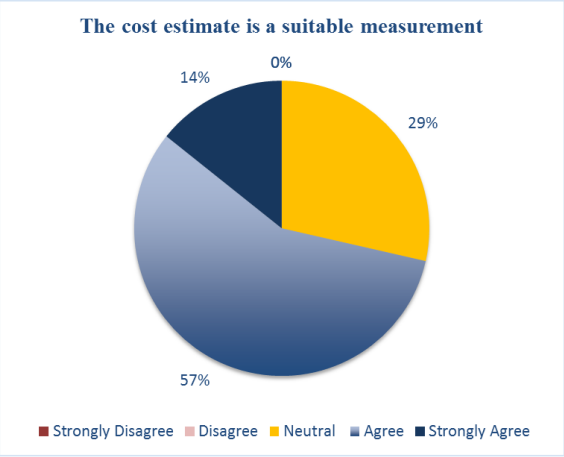
Table 9-7 Interviewee 7

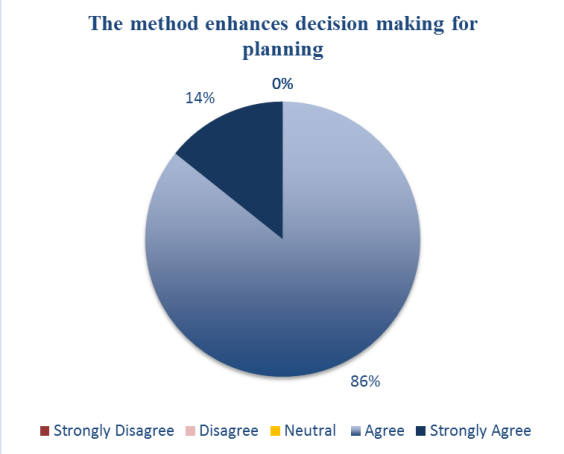
Interviewee Background	*Position*: Business Analyst *Field of Expertise:* Specialist in ERP and information systems *Experience in ERP system:* 5 Years
Q1) Any experience using impact analysis tool in your ERP system	Since we have small group people without having many dependencies in our system, also, we had some minor modifications to our ERP system, so we do not use any impact analysis tool for that.
Q2) Approaches for analysis of Impact analysis (e.g. expert judgment, previous historical data, and previous experience, etc.)	We do not have techniques or not following any methods during the change process. In case if there is change due to an error or change in the procedure mainly we discuss it with end user first and identify what need to change, and then we pass this information to developer from another company to provide with the exact information about the impact on the whole system.
Q3) Assessment through standard set of steps/activities or a process to trace the impact of modification	As mentioned, we do not have any standard procedure to deal with the change in our system. But the steps that involve in such a situation is first to look at the budgets and provide a time frame for implementations. Indeed, we need approval from our manager for implementing the change. In order to do that, we need all the information regarding on why we need to change and what is the consequence of the modification. This information is gathered from the discussion by the end-user and the development company that provide detailed information about the impact.
Q4) Related information reuse during the assessment i.e. dependency model, change documentation, historical change data	The only document that we are using during the process of modification is a change log that is a repository of all improvements to the system.
Q5) Importance of having structured approach and Tool for assessing the impact	For us as a small organisation, we are not experiencing with so many changes. Therefore, it is not much important for us to have an impact analysis tool due to a small group of people and departments. However, impact analysis tool can be very useful for those companies with thousand documentation and dependencies to assess the dependencies and impact of change.

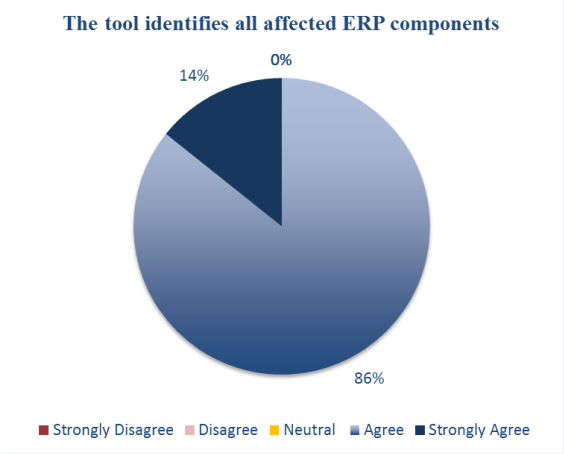
Experimental Study with ERP expert

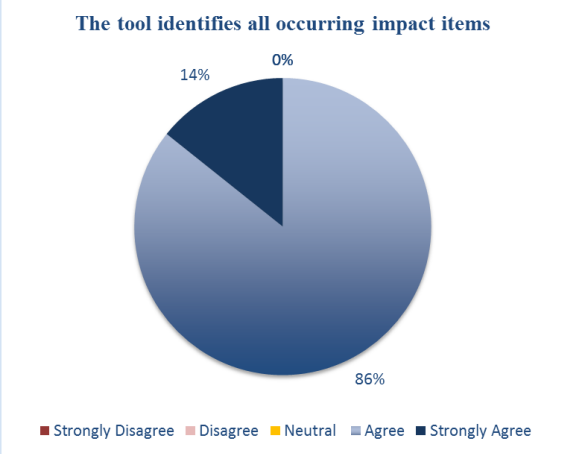
Functionality assessment

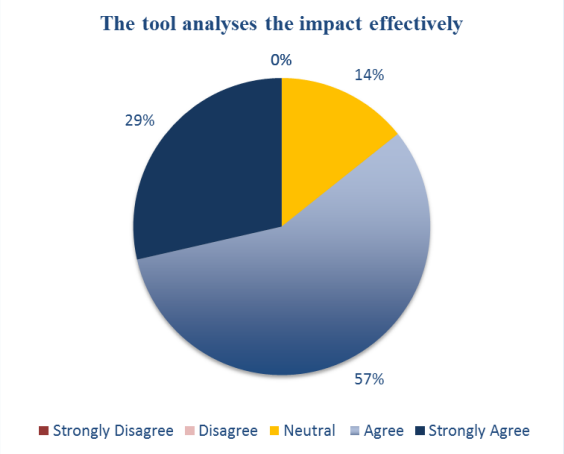


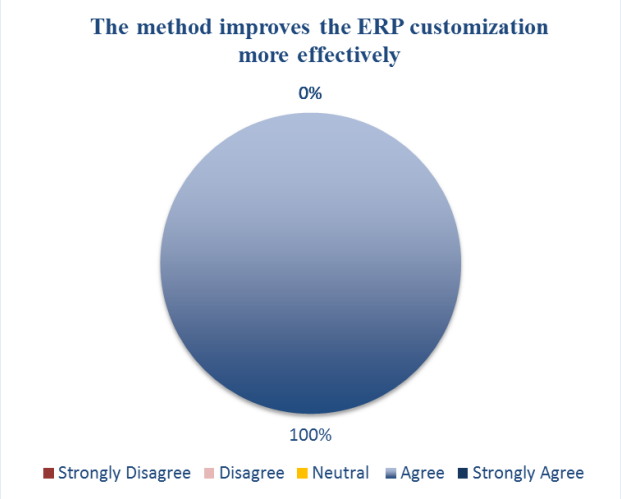


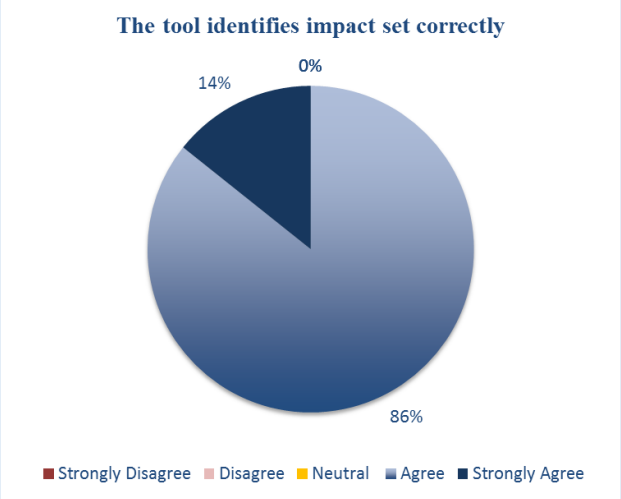


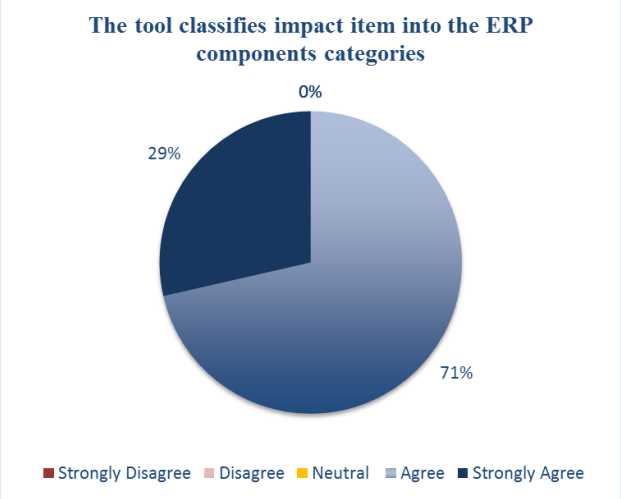




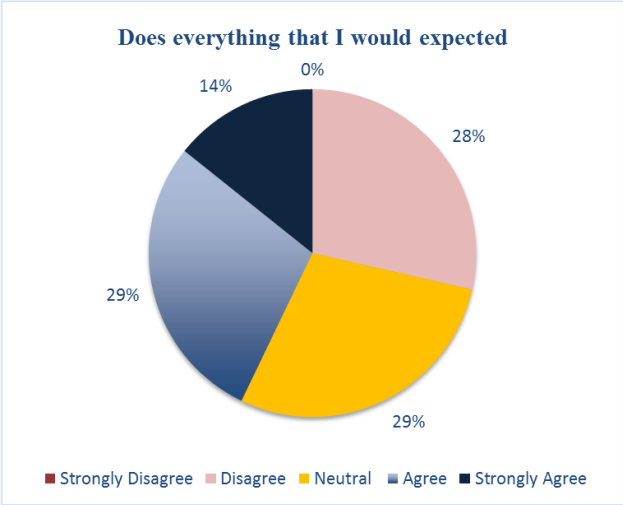


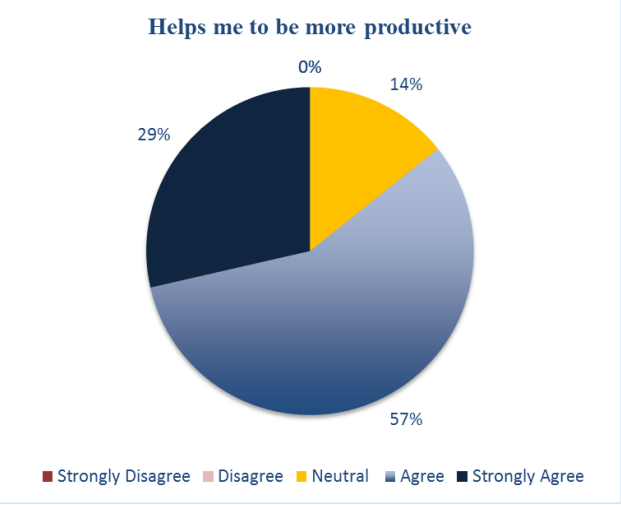

The method improves the ERP customization more effectively
0%
100%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

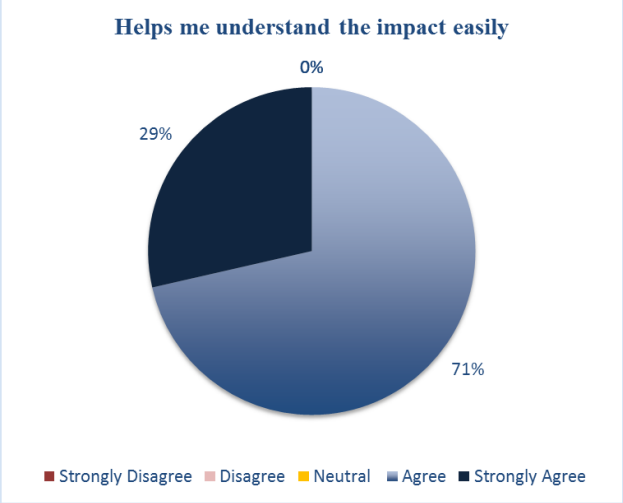

The tool identifies impact set correctly
0%
14%
86%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

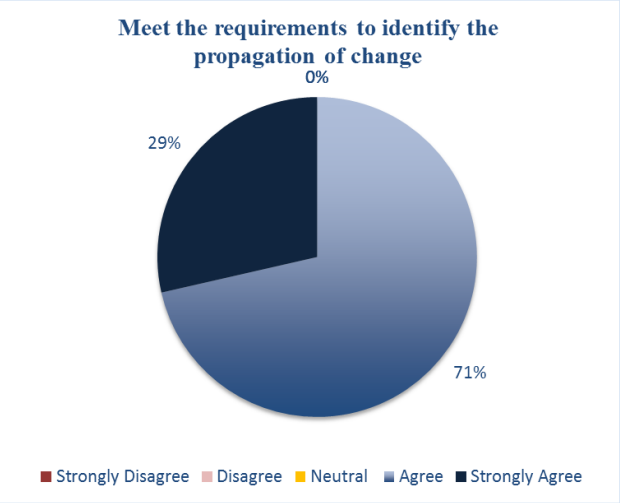

The tool classifies impact item into the ERP components categories
0%
29%
71%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

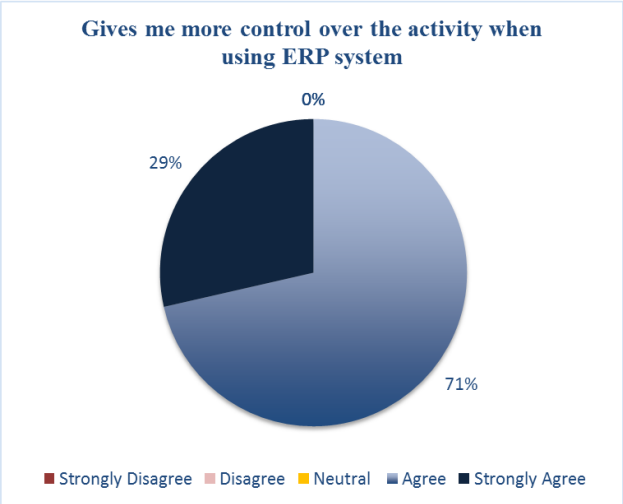
Usefulness Assessment

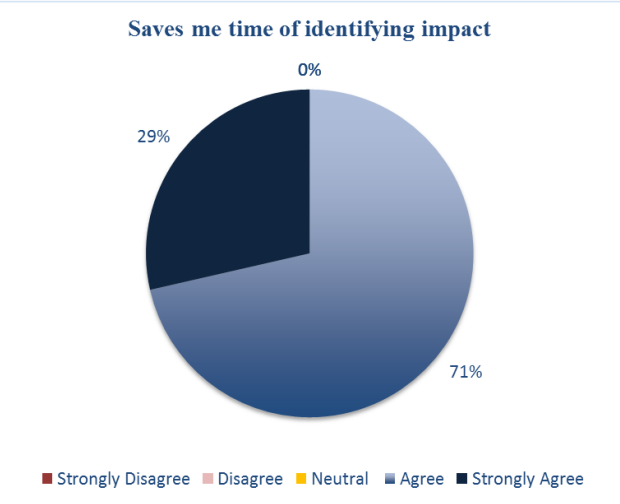


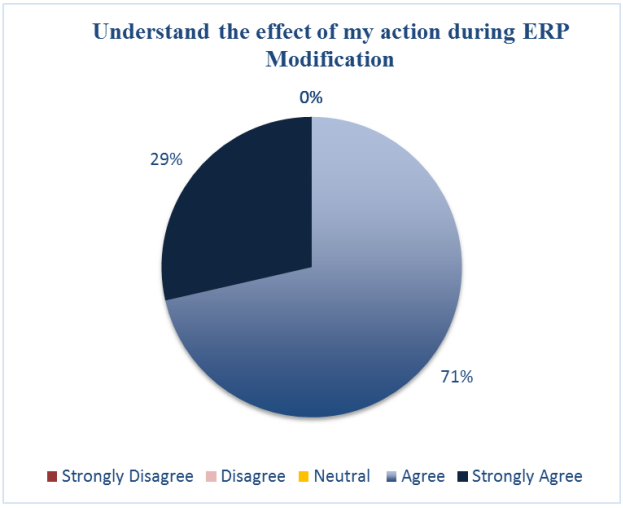


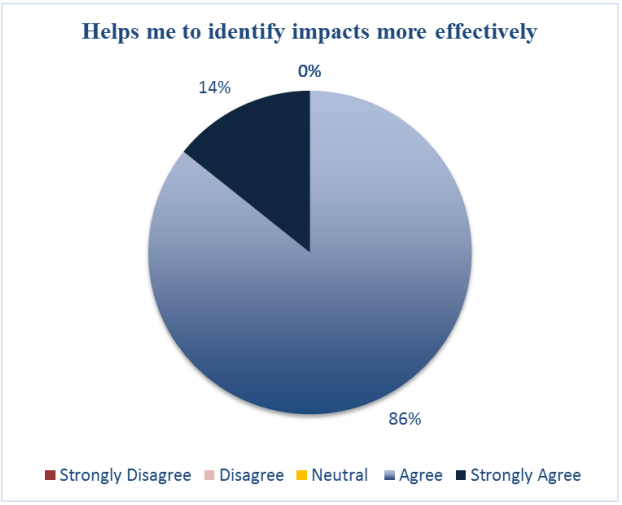




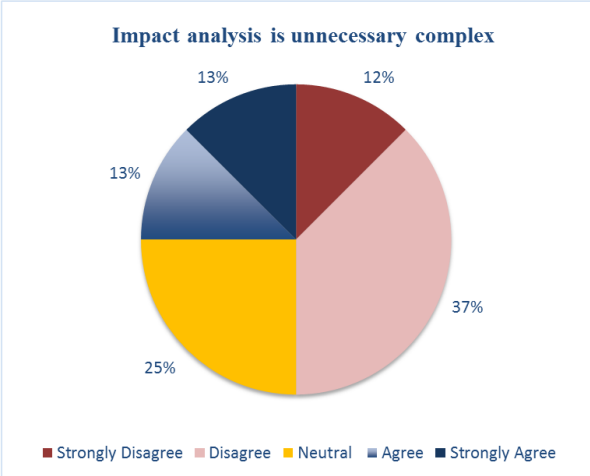


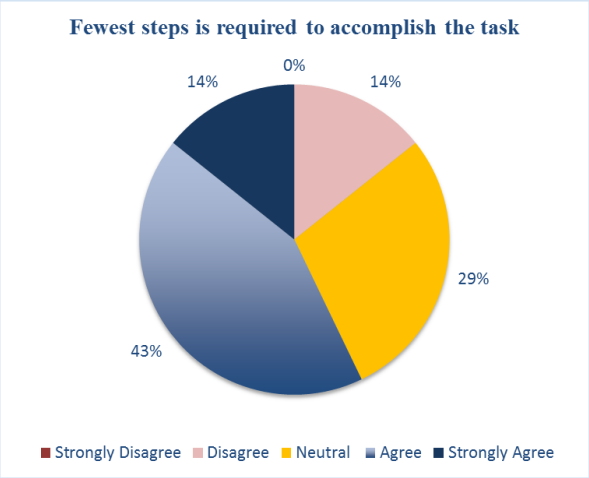


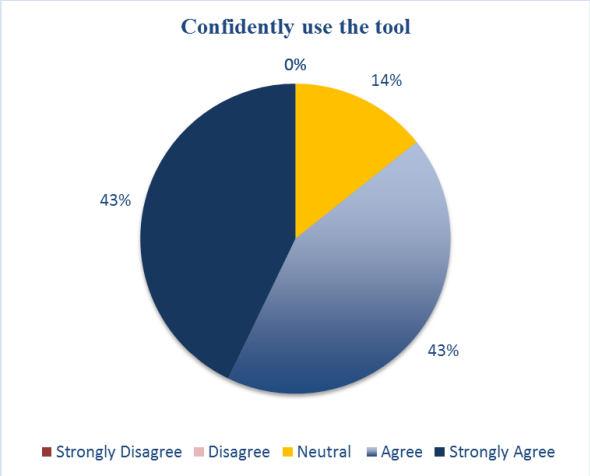

Understand the effect of my action during ERP Modification
0%
29%
71%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

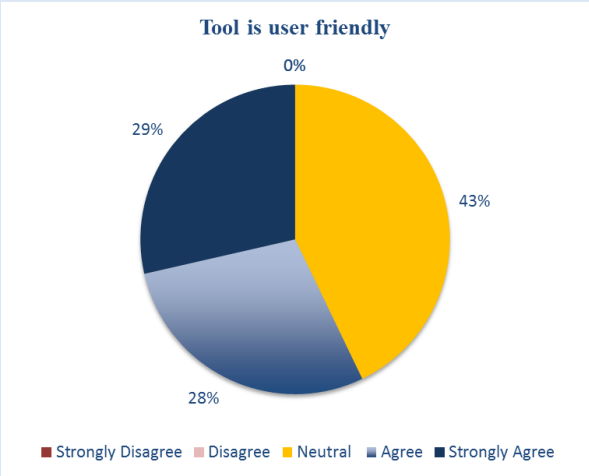

Helps me to identify impacts more effectively
0%
14%
86%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

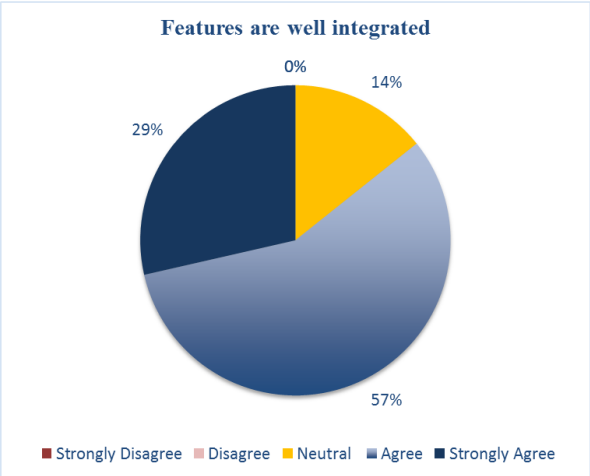
Ease of Use Assessment

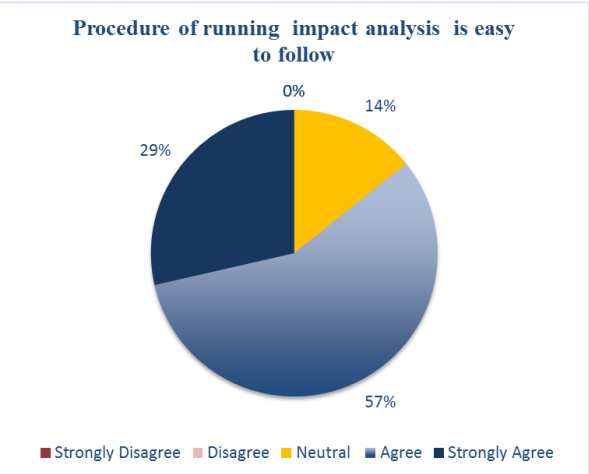


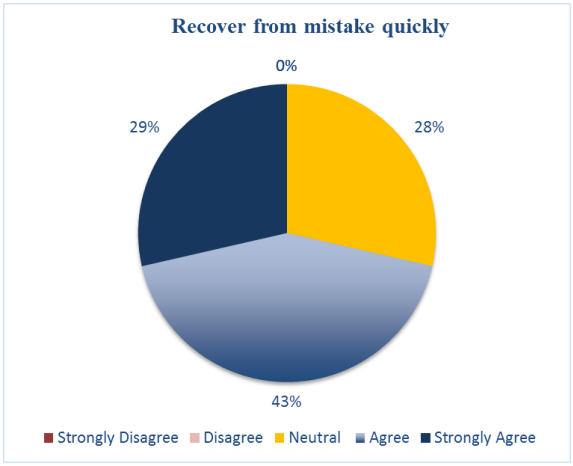




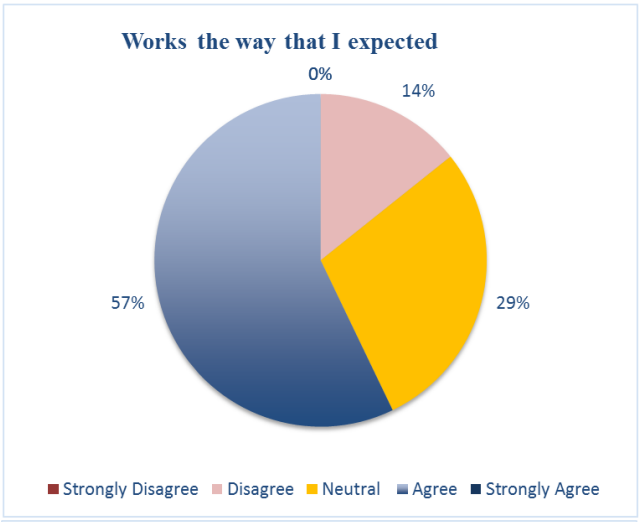


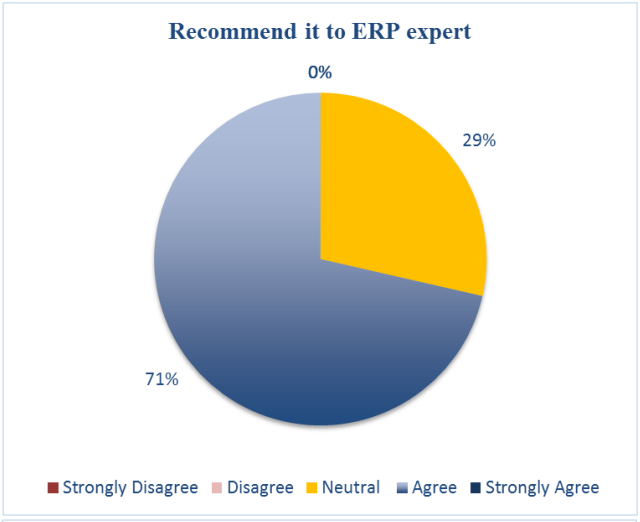


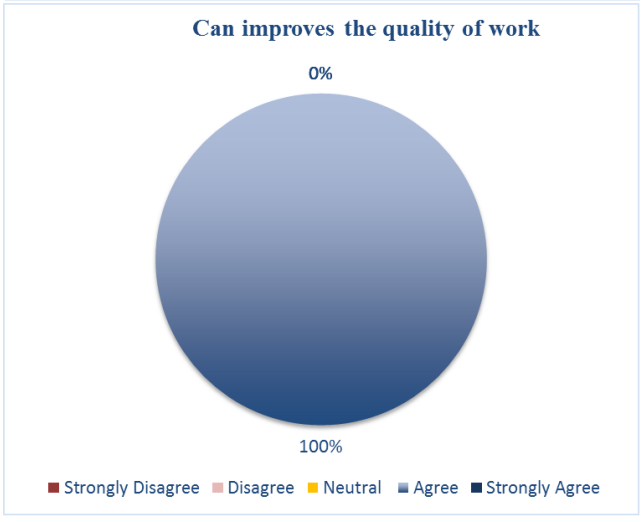


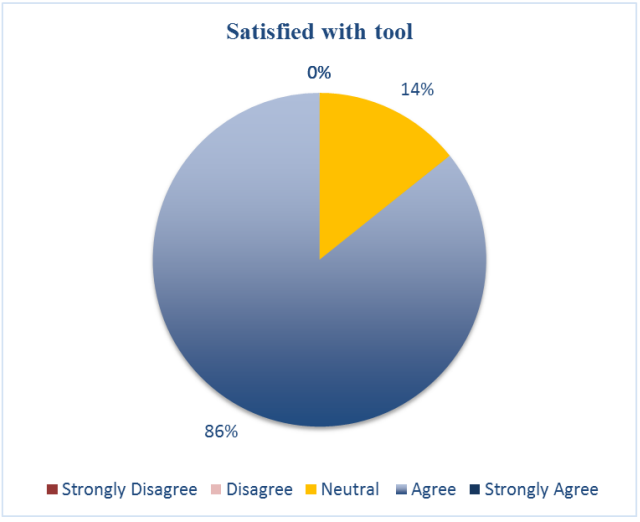


Satisfaction Assessment

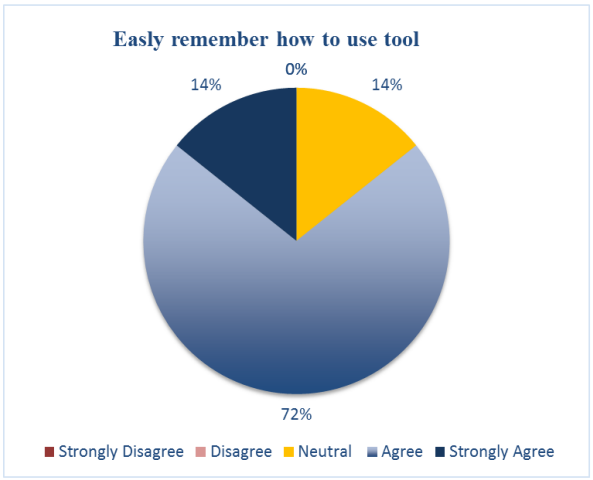


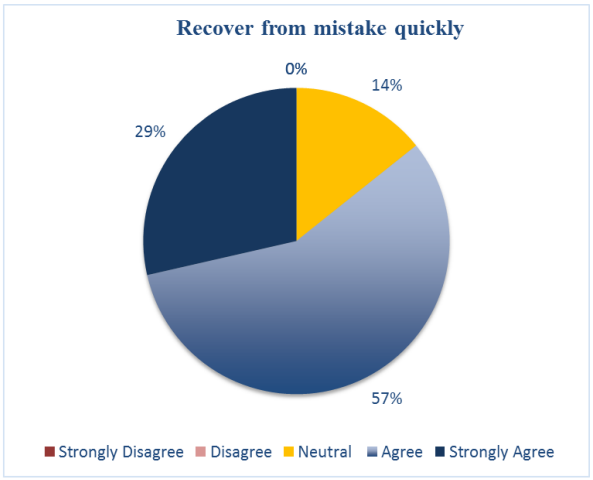


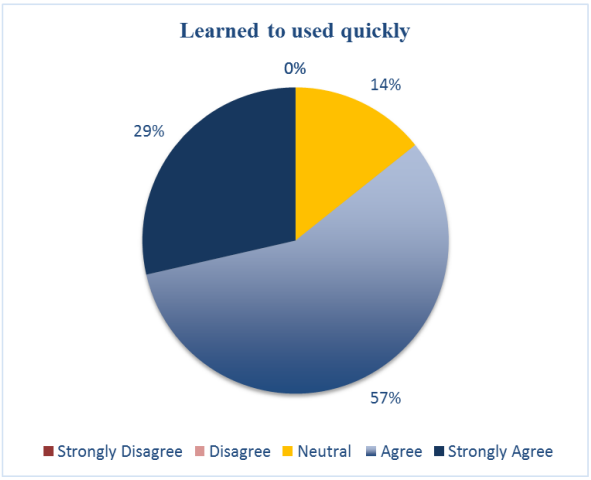




Ease of Learning Assessment

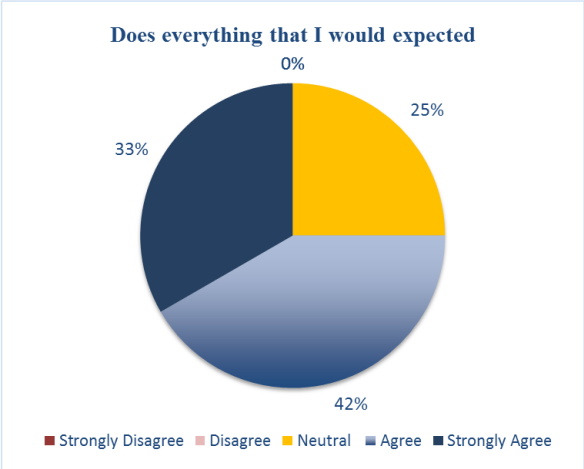


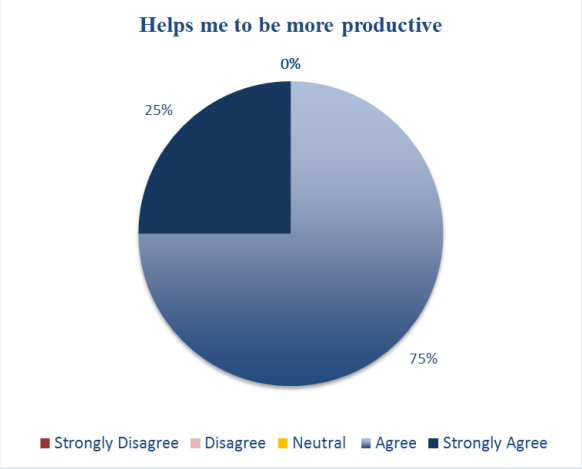


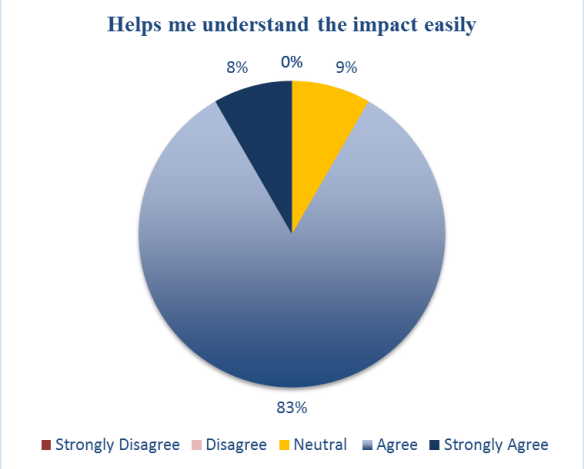


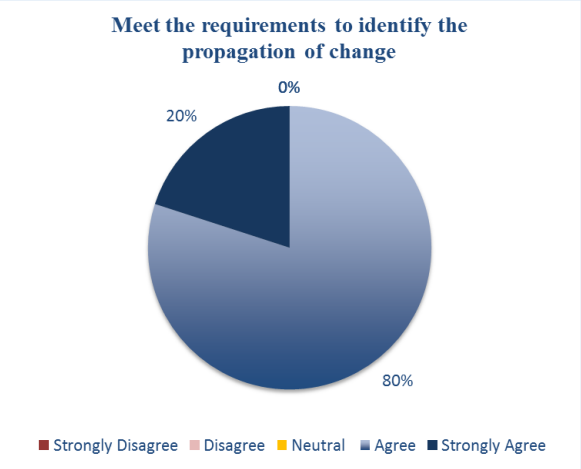
Experimental Study with Non-ERP expert

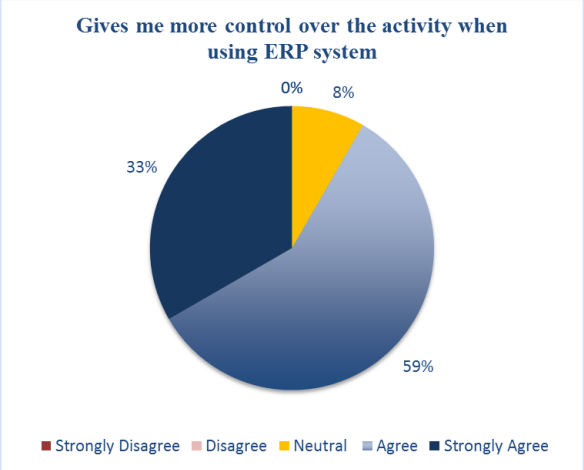
Usefulness Assessment

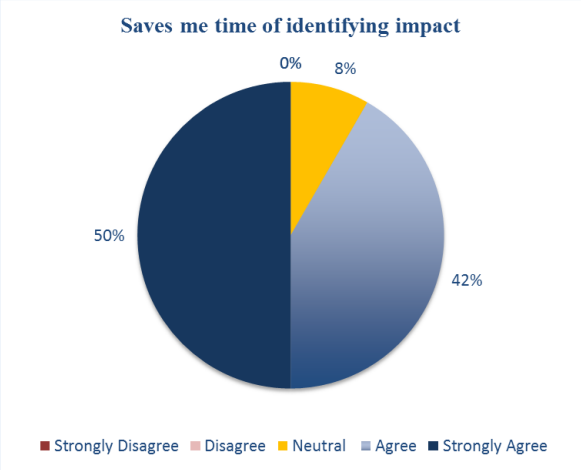


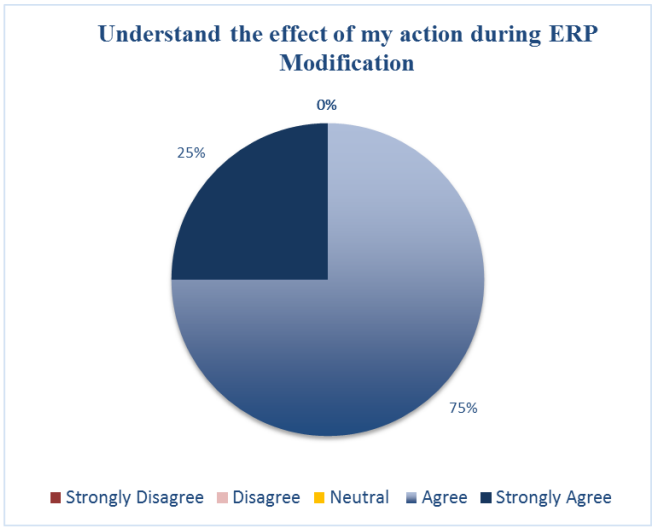


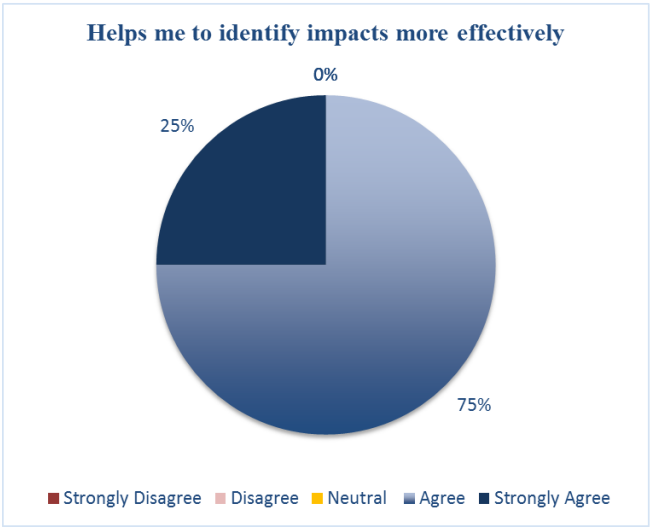




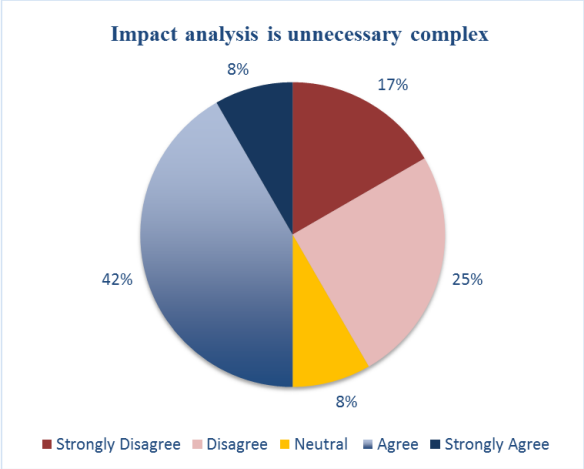


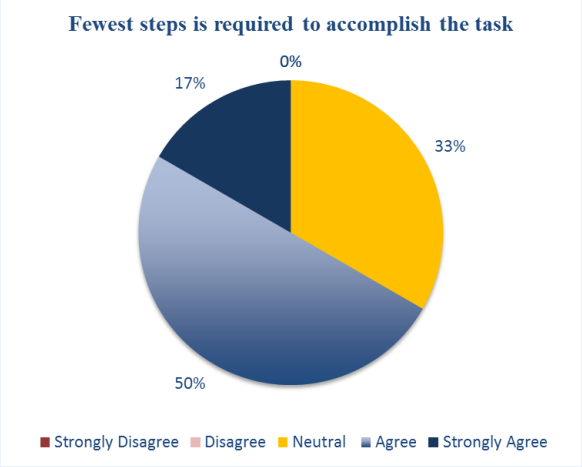


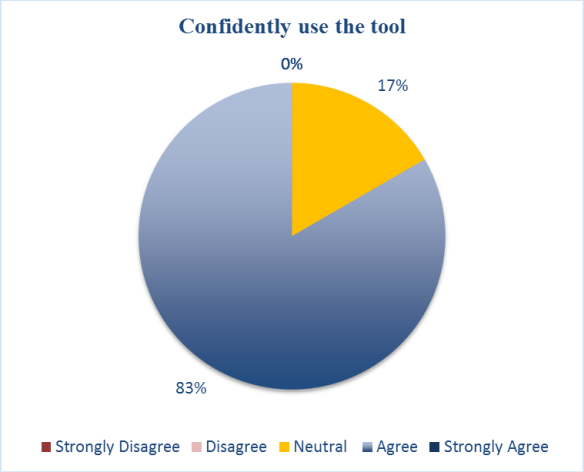

Understand the effect of my action during ERP Modification
0%
25%
75%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

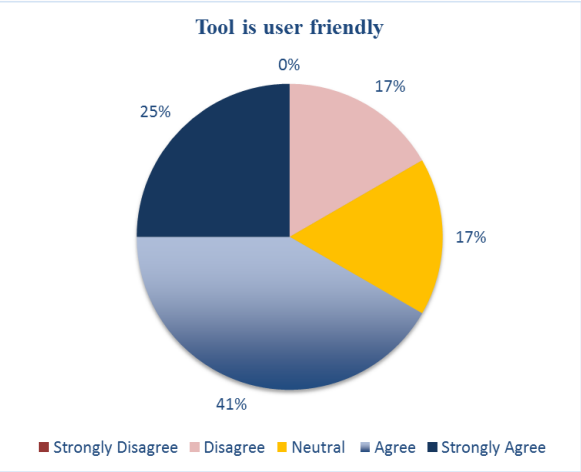

Helps me to identify impacts more effectively
0%
25%
75%
Strongly Disagree
Disagree
Neutral
Agree
Strongly Agree

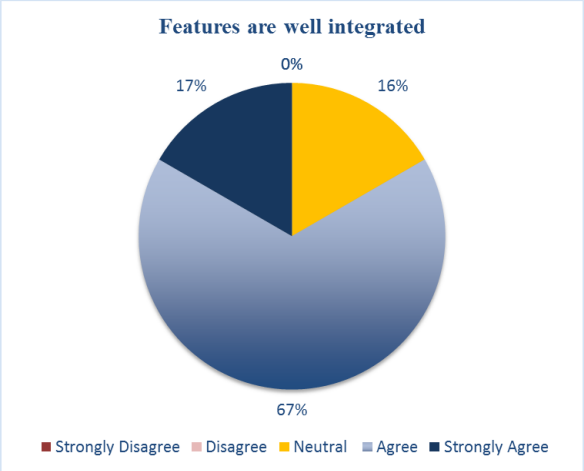
Ease of Use Assessment

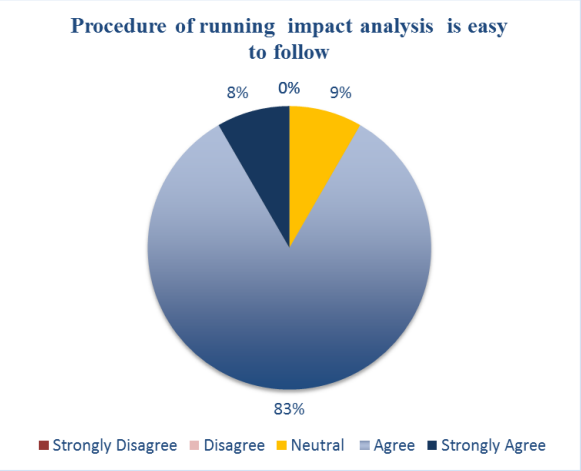


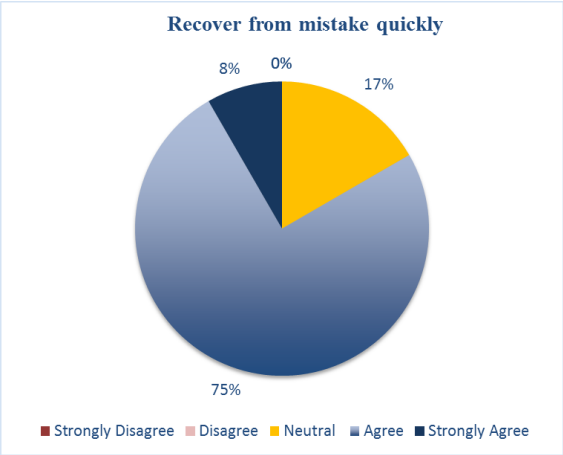




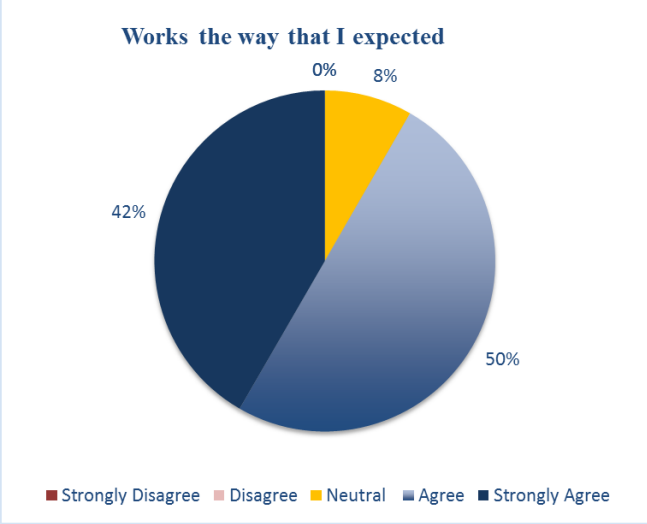


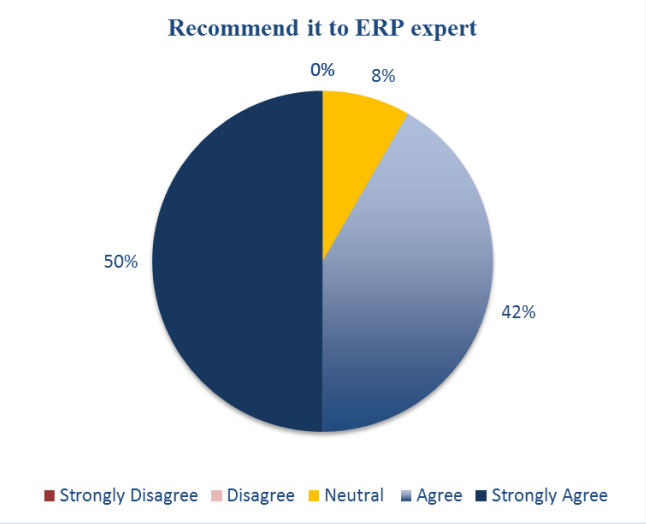


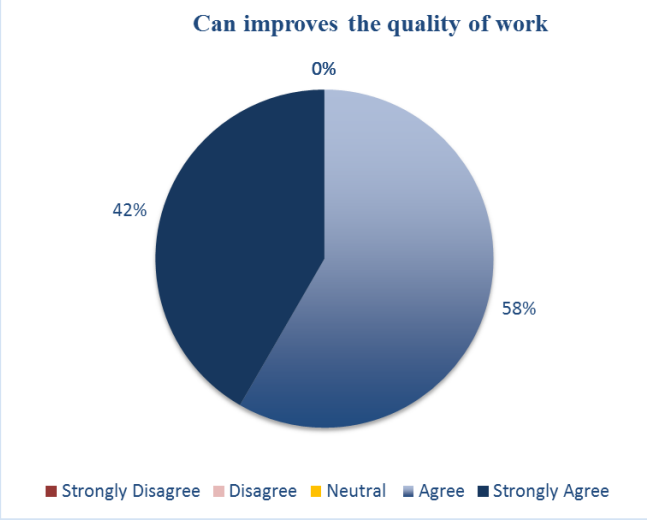


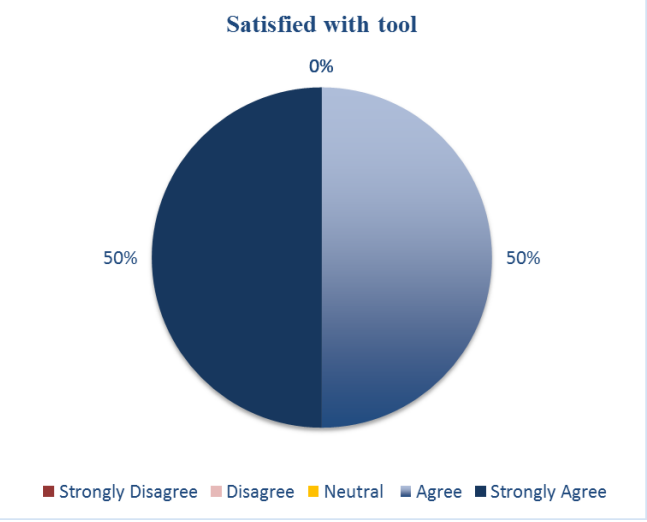
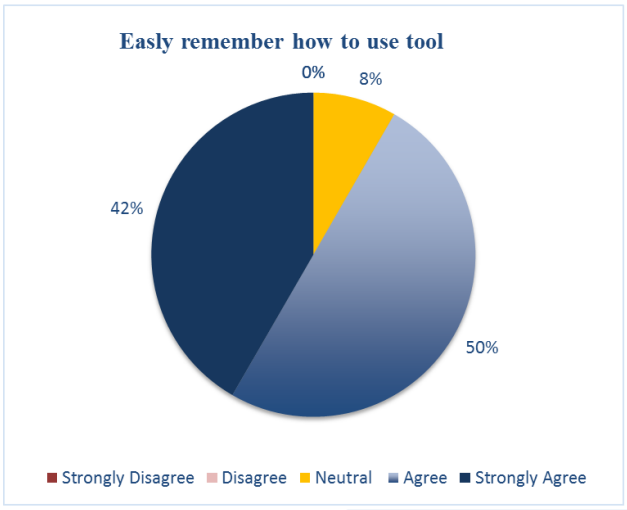
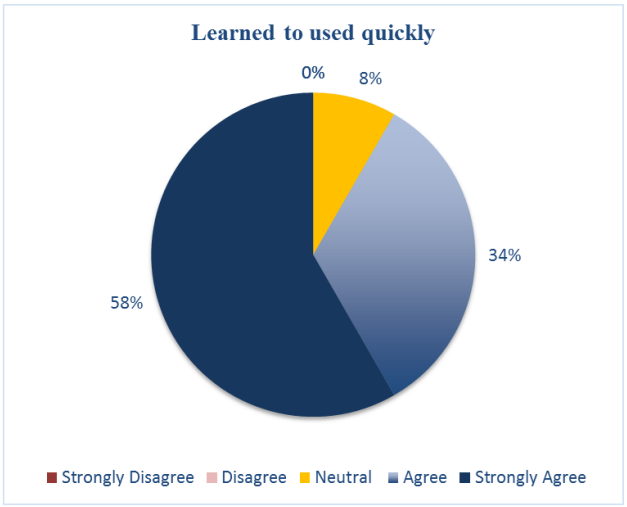
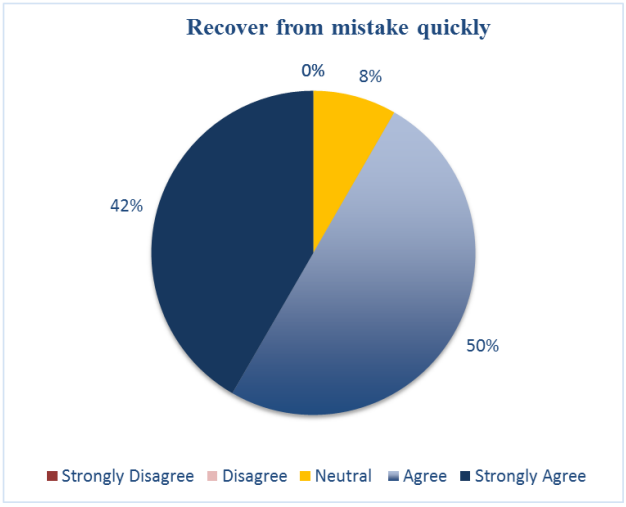


Satisfaction Assessment









Ease of Learning Assessment