A Multiple-Criteria Approach to Support Complex Decisions In Extreme Programming

A Thesis
Submitted to the Faculty of Graduate Studies and Research
In Partial Fulfilment of the Requirements
For the Degree of
Doctor of Philosophy
In
Software Systems Engineering
University of Regina

By
Abdulmajeed Abdulrhman Aljuhani
Regina, Saskatchewan
December, 2017

Copyright 2017: A. Aljuhani
Abdulmajeed Abdulrhman Aljuhani, candidate for the degree of Doctor of Philosophy in Software Systems Engineering, has presented a thesis titled, *A Multiple-Criteria Approach to Support Complex Decisions in Extreme Programming*, in an oral examination held on December 8, 2017. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner:       *Dr. Ralph Deters, University of Saskatchewan
Supervisor:              *Dr. Luigi Benedicenti, Adjunct
Committee Member:        Dr. Craig Gelowitz, Software Systems Engineering
Committee Member:        **Dr. Yasser Morgan, Software Systems Engineering
Committee Member:        Dr. Malek Mouhoub, Department of Computer Science
Chair of Defense:        Dr. Harold Weger, Department of Biology

*Via ZOOM
**Not present at defense
ABSTRACT

The software industry has been growing rapidly in recent decades, demonstrating an increasing need for software. There are several methods that can be used to design appropriate software to ensure high satisfaction levels. One of these is the agile methodology, which allows development teams to design projects in different iterations. This incremental design allows customers to provide feedback during the design stage, thereby increasing the possibility of delivering satisfactory projects.

Most customers do not know what they need at the beginning, which makes it difficult to define the system requirements. In the case of traditional software methods, it is costly to change the requirements after starting the development process. Therefore, software engineers try to find a method that allows them to start with undefined requirements, so they can more easily accept requirement changes during the development process. The agile method was introduced as an incremental method to develop software through a series of iterations. Extreme Programming (XP), the most successful agile method, allows developers to deliver valuable software to the customer. Extreme programming consists of twelve practices that structure the XP development cycle. Following these practices, all team members are able to work together. Extreme programming includes five major values, such as simplicity, feedback, etc., all of which help the development team to achieve a high level of customer satisfaction.

During the development cycle, team members encounter issues that require de-
decisions to be made, such as evaluating factors that affect the extreme programming life cycle and productivity. The Analytic Network Process (ANP) can be used to analyze complicated decisions and assist in achieving XP values. This research explains the application of the analytic network process to XP in order to solve conflict factors by addressing the decision making process. The ANP provides a structure to address solutions for given problems, which leads to a decision being made concerning that problem. In the ANP, dependencies among various criteria are considered, which makes it different from the Analytic Hierarchy Process (AHP). In pairwise comparisons, entered values reflect the relative effect among elements with respect to a control criterion. These entered values are based on the importance of each criterion.

In this research, there are nine areas related to XP practices that could potentially be improved by applying the ANP. The quality of the selected XP practices, which include planning game, simple design, refactoring, pair programming, and test-driven development, depend on decisions made by the XP team. These are ideal for ANP. The ANP can be applied in the planning game practice in several areas, such as ranking user stories estimation techniques and ranking user stories prioritization techniques. In pair programming, the ANP is used to select the appropriate matching pairs and for deciding the rules of matching pairs. In simple design, the ANP is used to select the best design tool. In refactoring, the ANP is used to rank the refactoring techniques based on the internal and external attributes. In testing, the ANP is applied to decide the level of automated testing and to rank software release indicators.
Acknowledgement

First of all, great thanks must be given to my supervisor, Dr. Luigi Benedicenti, for his cooperation during all phases of this research and particularly for his guidance, wisdom, patience and positive encouragement that enabled me to complete this thesis. I often benefited from his advice, specifically when researching new ideas. His positive outlook and confidence gave me confidence. It was an honor to work under his supervision and to learn from his experience in the field of software engineering. Without his guidance this thesis would not have been successfully completed.

My sincerest gratitude goes to my family for their support and patience. Without their support, I would not have been able to successfully finish this report.

In addition, I would like to thank all of my friends, especially Dr. Sultan Al-shehri, who supported and helped me to overcome challenges I encountered during this research.

Also, I would like to thank the Saudi Cultural Bureau in Canada and Taibah University in Saudi Arabia for their financial support during all these years.

Lastly, I would like to thank the committee members, Dr. Malek Mouhoub, Dr. Yasser Morgan, and Dr. Craig Gelowitz, for their time, support and valuable comments.
Dedication

I would like to dedicate this research to my mother for her lifelong support of me. Without her prayers and encouragement I would not have been able to complete this study. I would also like to dedicate this research to my father, whose confidence in me gave me the confidence to finish my PhD.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>i</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td>iii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td>iv</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xiii</td>
</tr>
<tr>
<td>ABBREVIATIONS</td>
<td>xiv</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Research Problem</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Motivation</td>
<td>5</td>
</tr>
<tr>
<td>1.3 Research Objectives</td>
<td>6</td>
</tr>
<tr>
<td>1.4 Methodology and Scope</td>
<td>6</td>
</tr>
<tr>
<td>1.5 Original Contributions</td>
<td>7</td>
</tr>
<tr>
<td>1.6 Organization of Thesis</td>
<td>8</td>
</tr>
<tr>
<td>2 Literature Review</td>
<td>10</td>
</tr>
<tr>
<td>2.1 Agile Methodology</td>
<td>10</td>
</tr>
</tbody>
</table>
# Table of Contents

1. **Introduction** ......................................................... 7
2. **Extreme Programming (XP)** ...................................... 14
   - 2.2.1 Extreme Programming Process .......................... 19
   - 2.2.2 Extreme Programming Practices ....................... 21
3. **The Analytic Network Process (ANP)** ....................... 26
4. **Why Analytic Network Process** ................................. 29
3. **Applications of The Analytic Network Process** .......... 32
   - 3.1 Technology Applications of the ANP .................... 32
   - 3.2 ANP Integration in Technology Applications .......... 50
   - 3.3 Applications of The ANP in Other Fields ............... 57
4. **Case Studies Set-up** ............................................. 58
   - 4.1 Methodology .................................................. 58
   - 4.2 Case Study Design ......................................... 59
     - 4.2.1 Research Questions in General Format ........... 59
     - 4.2.2 Research Propositions in General Format ........ 60
     - 4.2.3 Specific Research Questions and Propositions for Each XP Practice ............................................. 61
   - 4.3 Criteria for Interpretation .................................. 65
   - 4.4 Unit of Analysis ............................................. 66
   - 4.5 Data Collection and Sources ................................ 66
   - 4.6 Questionnaires ............................................... 67
   - 4.7 Semi-Structured Interviews ................................. 67
   - 4.8 Field Notes .................................................. 68
   - 4.9 Academic Case Study ....................................... 68
     - 4.9.1 Team Formation ........................................ 69
     - 4.9.2 Development Tools ..................................... 69
     - 4.9.3 Communication Tools .................................. 69
5 Applying ANP into Extreme Programming Practices

5.1 ANP in Planning Game

5.1.1 User Stories Overview

5.1.2 Features Influencing Requirements Prioritization

5.1.3 Applying ANP in Planning Game

5.1.4 First: Ranking the Prioritization Methods

5.1.5 Second: Ranking the Estimation Methods

5.2 ANP in Simple Design

5.2.1 Introduction

5.2.2 Selecting the Simple Design Tool

5.2.3 Proposed Criteria for Choosing the Design Method

5.2.4 ANP Structure for Design Tools

5.2.5 Pairwise Comparisons for the Design Methods

5.2.6 Design Tools Results Based on ANP Evaluation

5.2.7 Observations

5.3 ANP in Pair Programming

5.3.1 Introduction

5.3.2 Applying ANP in Pair Programming

5.3.3 First: Selecting the Best Matching Pairs

5.3.4 Second: Ranking the Rules for Matching Pairs

5.4 ANP in Refactoring

5.4.1 Introduction

5.4.2 Refactoring Techniques

5.4.3 Applying ANP in Refactoring
5.4.4 First: Ranking Refactoring Methods Based on Internal Quality Attributes ........................................... 146

5.4.5 Second: Ranking Refactoring Methods Based on External Quality Attributes ........................................... 159

5.5 ANP in Test-Driven Development ........................................... 174

5.5.1 Introduction ........................................... 174

5.5.2 Applying ANP in Test-Driven Development ........................................... 175

5.5.3 First: Automated Testing Decision ........................................... 176

5.5.4 Second: Ranking Release Indicators ........................................... 186

5.6 Analysis and Discussion ........................................... 196

5.6.1 Semi-Structure Interview ........................................... 196

5.6.2 Questionnaires ........................................... 200

5.6.3 Logic Linking of the Data to the Research Questions and Propositions ........................................... 202

5.6.4 Interpretation of the results ........................................... 202

5.7 Validity ........................................... 207

5.7.1 Construct Validity ........................................... 207

5.7.2 Internal Validity ........................................... 208

5.7.3 External Validity ........................................... 208

5.7.4 Reliability ........................................... 208

6 Conclusion and Future Work ........................................... 209

6.1 Summary ........................................... 209

6.2 Future Work ........................................... 213

REFERENCES ........................................... 215
# LIST OF FIGURES

2.2 The analytic network process structure [2] ........................................... 28
2.3 The Super-matrix of a network [2] ....................................................... 29

5.1 ANP network for ranking the prioritization methods .................................. 84
5.2 The importance of the criteria by Team 1 .............................................. 91
5.3 ANP structure for selecting the best estimation method .............................. 99
5.4 The importance of the criteria by Team 1 .............................................. 105
5.5 ANP network for selecting the design method ........................................ 111
5.6 The importance of the criteria by Team 1 .............................................. 117
5.7 ANP network for selecting the best pairs ............................................. 125
5.8 The importance of the criteria by Team 1 .............................................. 130
5.9 ANP network for ranking the rules of matching pairs ............................... 135
5.10 The importance of the criteria by Team 1 in the rules of matching .......... 140
5.11 ANP network for ranking refactoring techniques based on the internal attributes .................................................. 149
5.12 The importance of the internal attributes for the refactoring patterns by Team 1 .................................................. 156
5.13 ANP network for ranking refactoring techniques based on the external attributes .................................................. 161
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>ANP fundamental scale developed by Saaty [3]</td>
<td>28</td>
</tr>
<tr>
<td>2.2</td>
<td>Random Index [2]</td>
<td>29</td>
</tr>
<tr>
<td>3.1</td>
<td>Summary of the ANP Applications in Technology with their purposes</td>
<td>49</td>
</tr>
<tr>
<td>3.2</td>
<td>Summary of the integration of ANP in Technology Applications</td>
<td>56</td>
</tr>
<tr>
<td>3.3</td>
<td>Summary of the ANP in various Applications</td>
<td>57</td>
</tr>
<tr>
<td>4.1</td>
<td>User stories and tasks for iteration 1</td>
<td>72</td>
</tr>
<tr>
<td>4.2</td>
<td>User stories and tasks for iteration 2</td>
<td>73</td>
</tr>
<tr>
<td>4.3</td>
<td>User stories and tasks for iteration 3</td>
<td>73</td>
</tr>
<tr>
<td>4.4</td>
<td>User stories and tasks for iteration 4</td>
<td>73</td>
</tr>
<tr>
<td>4.5</td>
<td>User stories and tasks for iteration 5</td>
<td>73</td>
</tr>
<tr>
<td>4.6</td>
<td>Team 1 technical information for each iteration</td>
<td>74</td>
</tr>
<tr>
<td>4.7</td>
<td>Team 2 technical information for each iteration</td>
<td>74</td>
</tr>
<tr>
<td>5.1</td>
<td>Prioritization methods ranking by Team 1</td>
<td>90</td>
</tr>
<tr>
<td>5.2</td>
<td>Prioritization methods ranking by Team 2</td>
<td>91</td>
</tr>
<tr>
<td>5.3</td>
<td>The importance of the criteria by Team 2</td>
<td>91</td>
</tr>
<tr>
<td>5.4</td>
<td>Prioritization methods with respect to each criterion for Team 1</td>
<td>94</td>
</tr>
<tr>
<td>5.5</td>
<td>Criteria weights with respect to each alternative for Team 1</td>
<td>94</td>
</tr>
<tr>
<td>5.6</td>
<td>Estimation methods ranking for Team 1</td>
<td>104</td>
</tr>
<tr>
<td>5.7</td>
<td>Estimation methods ranking by Team 2</td>
<td>105</td>
</tr>
</tbody>
</table>
5.8 The importance of the criteria by Team 2 ........................................ 105
5.9 Estimation methods with respect to each criterion for Team 1 .......... 107
5.10 Criteria weights with respect to each alternative for Team 1 .......... 107
5.11 Design methods ranking for Team 1 ............................................. 116
5.12 Design methods ranking by Team 2 ............................................ 117
5.13 The importance of the criteria by Team 2 .................................... 117
5.14 Design methods with respect to each criterion for Team 1 .......... 119
5.15 Criteria weights with respect to each design tool for Team 1 ....... 119
5.16 Matching pairs ranking for Team 1 ............................................. 130
5.17 Matching pairs ranking by Team 2 ............................................ 131
5.18 The importance of the criteria by Team 2 .................................... 131
5.19 Ranking pairs with respect to each criterion for Team 1 .......... 132
5.20 Criteria weights with respect to each pair for Team 1 ............... 132
5.21 Matching pairs ranking for Team 1 ............................................. 140
5.22 Rules of pairing ranking by Team 2 ............................................ 140
5.23 The importance of the criteria by Team 2 in the rules of pairing ... 141
5.24 Pairing options with respect to each criterion individually for Team 1 142
5.25 Criteria weights with respect to each pairing option for Team 1 .... 142
5.26 Ranking the refactoring patterns based on internal attributes by Team 1 156
5.27 Refactoring techniques ranking by Team 2 ............................... 156
5.28 The importance of the criteria by Team 2 .................................... 157
5.29 Refactoring techniques with respect to each criterion for Team 1 .... 158
5.30 Criteria weights with respect to each refactoring technique for Team 1 158
5.31 Ranking the refactoring patterns based on external attributes by Team 1 168
5.32 Refactoring methods ranking by Team 2 .................................... 169
5.33 The importance of the criteria by Team 2 .................................... 170
5.34 Refactoring techniques with respect to each criterion for Team 1 .... 171
ABBREVIATIONS

#LOC Number of Lines of Code

AHP Analytic Hierarchy Process

AM Agile Modeling

ANN Artificial Neural Network

ANP Analytic Network Process

AOP Aspect Oriented Programming

AUP Agile Unified Process

C.R Consistency Ratio

CAD Computer Aided Design

CAE Computer Aided Engineering

CAM Computer Aided Manufacturing

CCMN Cloud Computing Modeling Notation

CRC Class Responsibility Collaboration

DSDM Dynamic Systems Development Method

ERP Enterprise Resource Planning
GUI  Graphic User Interface

IaaS  Infrastructure as a Service

LOCC  Lines of Code for Class

MCDM  Multi-Criteria Decision Making

NOC  Number of Children

NOF  Number of Fields

NOM  Number of Methods

NOTC  Number of Test Cases

PaaS  Platform as a Service

PDM  Product Data Management

PDP  Product Development Process

PLM  Product Lifecycle Management

POS  Point-Of-Sale

PTC  Parametric Technology Corporation

QoS  Quality of Service

RBAC  Role Based Access Control

ROI  Return On Investment

SaaS  Software as a Service

SLA  Service Level Agreement

TDD  Test-Driven Development
TOGAF  Open Group Architecture Framework

TPS  Transaction Processing System

UGS  Unigraphics Solutions

UML  Unified Modeling Language

XP  Extreme Programming
Chapter 1

INTRODUCTION

Making correct decisions takes time and money, illustrating a need for techniques and tools that help decision makers. In the software industry, there are several critical issues that require making appropriate decisions in order to avoid negative consequences. The difficulty of decision making, however, rests in choosing among different alternatives with large numbers of uncertainties, and as Anderson noted, “decisions become more difficult when multiple factors are involved” [4]. Decision makers, therefore, find it difficult to select among various alternatives. Furthermore, increasing the number of alternatives increases the complexity of making decisions. However, these alternatives also represent solutions to a problem and, therefore, problems should be structured and defined precisely in order to prioritize appropriate solutions.

Furthermore, decision-making processes can have different definitions from different perspectives. For example, according to Fulop [5] “decision making is the study of identifying and choosing alternatives based on the values and preferences of the decision maker. Making a decision implies that there are alternative choices to be considered, and in such a case we want not only to identify as many of these alternatives as possible but to choose the one that best fits with our goals, objectives, desires, values, and so on” [5].
According to the above, therefore, decision making means starting with identifying a problem and outlining the desired goals that are to be achieved. Put another way, identifying the problem should describe all conditions and factors that are involved in the problem as well as describing the initial situation and desired targets. This stage is known as the problem statement which “must however be a concise and unambiguous written material agreed by all decision makers and stakeholders” [5]. Another stage in decision-making is specifying requirements. This means that any solution should meet these requirements in order to be acceptable, and “for any possible solution it has to be decided unambiguously whether it meets the requirements or not. We can prevent the ensuing debates by putting down the requirements and how to check them in a written material” [5]. Determining alternatives is another phase in decision making and these alternatives should exhibit various approaches to transform initial situations into desired targets. By checking the alternatives one by one, a decision can be selected that meets identified requirements.

In addition, determining appropriate criteria or attributes that are chosen based on targets is also a significant stage in decision making. These criteria must be selected carefully because choosing appropriate alternatives depends on them. As a result, the entire decision-making process depends on them.

Another decision-making stage is choosing an appropriate decision-making tool. There are a large number of tools that can be used to solve problems; however, choosing an appropriate tool depends on the complexity of the problem and the experience of decision makers.

1.1 Research Problem

Because of the complexity of software development, there is a need for more powerful decision-making tools that support the development process [6]. Extreme Program-
Extreme Programming (XP) is an agile method that has popularity in developing software, and several studies show that in order to extend the extreme programming scope with larger projects, there is more need for decision-making tools during the iteration cycle. For example, Drury et al. [7] organized a study that included 43 agile experts in order to observe the decision-making process and its obstacles during the development cycle. The result of the study showed that conflicting priorities were a major challenge in the decisions [7].

There are several practices in extreme programming, and each practice contains different issues that affect the decision making process. Therefore, the appropriate procedure is to study each practice individually, and come up with the possible solutions for the conflicts that affect the practice. Thus, the Analytic Network Process (ANP) is used to enhance the decision making process in several extreme programming practices. Planning game, simple design, testing, refactoring, and pair programming are XP practices, which include activities that require a decision to be made.

Various researchers studied several factors that affect the decision-making process. For example, Tessem [8] interviewed four software organizations in order to study the effect of experience in the decision-making process. The findings of the study showed that developers’ perspectives are affected by the experience factor, and developers, who have more experience in the selected organizations, have more influence in the decision-making process. In addition, Tessem [8] highlighted that “agile methodologies are supposed to increase developers’ say in not only the design and solution but also in how projects are organized, what tasks to priorities, and not least, the selection of work tasks in the daily work” [8]. This reflects a major issue, which is that most decisions are made in an insufficient way such as through experience or voting. This has a negative consequences, and as [9] states, “the downside of using voting as a decision making mechanism is that someone (or some group of someone) will lose [sic]. The win/lose mechanism can create a divide within the team which will pull
it apart over time or generate passive aggressive behavior” [9]. Furthermore, in [10] the authors have organized several observations and interviews within a number of agile teams in different countries, in order to figure out how decisions are made by the team members. The findings of the study exhibited that the team members did not use a linear process to evaluate their decisions. As [10] explained, “often, people’s experience drives their individual and group decisions and agile teams do not identify and evaluate a series of options, as the linear process outlines” [10].

Since extreme programming is an iterative and incremental approach, responding to quick changes should be considered during decision-making processes. Ignoring such consideration might lead for problems to the team members as [9] states, “teams can struggle with creating a consensus, become passionately split on an idea or simply struggle to understand the options that are appropriate. Each team needs a pallet of decision-making tools available (or a coach with those techniques) or risk failing to make good decisions or decisions in a timely manner” [9]. There may be several reasons for failure in agile projects, such as lack of team members’ communication, and lack of exchanging developers’ perspectives. In [11], the authors state, “almost every challenge in building great software is, in some way, a people problem. That challenge may be communicating effectively, dealing with the unpredictability of moods and motives, or figuring out how to harness people’s desire to do the right thing for the team and the project” [11]. Moczar [12] addressed three defects that might lead to failure in agile projects, and he explained that changing over following a plan is a major defect. Moczar stated that making a big number of changes comes with its cost, and with continuing to change the plan, the project code will keep changing and leading to small errors that accumulate and result in big defects in the project [12].

In this thesis, there are several research problems that are addressed. These are:

- Uncertainty in determining the factors that influence the decision-making process in extreme programming.
• Complexity of selecting the right alternative when several alternatives are available.

• Depending only on certain decision-making methods, such as expert opinion, has a negative effect on extreme programming values such as communication.

• There is insufficient guidance to support the different evaluations of extreme programming members during the software process.

• There is insufficient structure based on a scientific approach in order to provide a framework for various activities in different practices.

Regarding these previous points, the Analytic Network Process (ANP) is introduced to extreme programming in order to optimize the decision-making process in several extreme programming practices.

1.2 Motivation

Extreme Programming (XP) consists of several practices, which help in providing value for XP customers and is considered a popular agile method for software design. In addition, XP provides a kind of environment that gathers customers, developers, and managers as team members. XP contains five main pillars, which are simplicity, respect, courage, feedback, and communication. XP process is based on an incremental theory, which allows developers to deliver a product incrementally during the development process with a high level of maintaining quality [11]. In this research, the Analytic Network Process (ANP) might be used to optimize the decision-making procedure in the XP process by supporting the XP values and assisting the achievement of XP’s objectives. The ANP has been applied in several software applications with different purposes such as evaluating different models, selecting one of various alternatives, and prioritizing alternatives and criteria. This thesis exhibits the application
of Analytic Network Process to several practices in the XP process.

1.3 Research Objectives

The goal in this research is to apply the Analytic Network Process to Extreme Programming in order to assist making complicated decisions. This will allow us to make decisions based on a scientific approach and allow us to avoid random decisions. The objective is to emphasize various features that influence each practice and take a measure of these features using ANP. In addition, using the ANP, alternatives and criteria are structured in terms of clusters in order to help decision makers in facing selecting, prioritizing, or evaluating problems. This research aims to find a framework of using the analytic network process in software processes such as extreme programming. Our expectations in using ANP in XP practices are to provide improvements to the XP process, such as enhancing team communication and increasing members productivity at a high quality level.

The need for such research appears from various issues, such as ranking, evaluating, or selecting required structured decision-making. Therefore, in this research, all team members will be encouraged to engage in the related decisions.

1.4 Methodology and Scope

One of the most significant research methodologies in software engineering is case study. According to Layman, "case studies are valuable because they involve factors that staged experiments generally do not exhibit, such as scale, complexity, unpredictability, and dynamism". For this reason, the case study was selected as the research methodology and a case study setting makes up the case study analysis. Two teams of participants will develop XP project. One team will use the analytic network process and the other team will follow the traditional way of designing an
XP project.

There are 12 practices in extreme programming, and some of them are descriptive practices; therefore, these descriptive practices are not included in this study. In this research, the focus is on the main practices, such as pair programming, simple design, testing, planning game, and refactoring. The ANP is used at least once with these practices, and multiple times with some of them.

Despite Extreme programming being introduced into several large-scale projects, it is still more effective within small and medium size software projects. Increasing the project scale might decrease the XP efficiency, and there will be no benefits of applying the analytic network process in extreme programming. Therefore, in this research, we are introducing the analytic network process into extreme programming development within a small scale in order to increase the efficiency of XP in such projects.

1.5 Original Contributions

This research aims to present a framework to the extreme programming team based on the ANP that might be introduced into various XP practices. The provided framework will introduce several solutions for different problems that occur routinely during the development cycle, such as decisions, selecting, and ranking.

Depending only on the voice of the most experienced developer or expert manager might lead to a misunderstanding within the development team. The ANP will allow everyone to participate in the decision process. Applying the ANP in the XP life cycle is intended to help identify various objective and subjective features that influence XP practices. The contributions of this thesis can be outlined as follows:

1. Investigating the alternatives and criteria that influence the decisions of extreme programming practices.
2. Building a framework based on the ANP to structure decision-making problems in extreme programming practices.

3. Proposing the analytic network process into nine fields that relate to extreme programming practices.

4. Studying the pros and cons of applying the ANP in an XP development.

5. Specifying the ranking and decision fields in five extreme programming practices and introducing the following:

(a) ANP ranking model for prioritization methods. Published in [14].

(b) ANP ranking model for estimation methods. Published in [15].

(c) ANP selecting model for the design tools.

(d) ANP evaluating model for the roles of pair matching.

(e) ANP selecting model for the selection of the best pairs.

(f) ANP ranking model for the refactoring techniques based on internal quality attributes. Published in [16].

(g) ANP ranking model for the refactoring techniques based on external quality attributes.

(h) ANP evaluating model for deciding the automated level of testing.

(i) ANP ranking model for TDD release indicators.

1.6 Organization of Thesis

This thesis consists of six chapters. Chapter 1 shows an introduction along with issues in decision-making in an XP development. Chapter 1 also introduces the motivations, objectives, methodology, and contributions of this thesis.
Chapter 2 is a literature review of fundamental information for agile methodology in general, and introduces the XP process, practices, and values. Chapter 2 also describes the analytic network process. Defining the differences between the analytic hierarchy process and the analytic network process concludes this chapter.

Chapter 3 introduces the applications of the ANP and shows the various purposes of using ANP in software development. This chapter also presents some examples of integrating the ANP with different decision-making tools.

Chapter 4 explains the case study design conducted in this research and contains different information such as duration of the study, the students' background, and technical and communication tools. This chapter shows information about the XP project developed during the study.

Chapter 5 describes ANP based framework design for five XP practices, which are planning game, simple design, pair programming, re-factoring, and test-driven development. The results of using ANP in XP practices, and the students' feedback are described in this chapter.

Chapter 6 draws the final conclusion of the thesis, and discusses future work for research exploration.
Chapter 2

Literature Review

This chapter consists of two parts. The first part is about the history of agile method and its principles, and extreme programming process and practices. The second part is an introduction to the analytic network process (ANP).

2.1 Agile Methodology

In software development, defining system requirements is a significant step to begin the development process. Unfortunately, most of the customers do not know what they need at the beginning, which makes it difficult to define the system requirements. With traditional software methods, changing the requirements after stating the development process is costly. Therefore, software engineers tried to find a method that allows them to start with un-defined requirements, and accept that requirements change during the development process. Agile method has introduced as an incremental method that is used to develop software based on iterations [17]. The agile method focuses on deliverable software more than writing documentation [17]. In early 2001, a group of agile supporters gathered and came up with an agile manifesto, which structures the different type of agile methods. The manifesto is as follows: “We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

- Individuals and interaction over process and tools,
- Working software over comprehensive documentation,
- Customer collaboration over contract negotiation,
- Responding to change over following a plan.

That is, while there is a value in the items on the right, we value the items on the left more” [17].

The significance of the manifesto comes from making agile methodologies different from other traditional software methods. The agile methods consider the importance of people and their community rather than emphasizing processes and tools, as well as individuals and interaction over process and tools. Another matter is that agile teams focus on delivering working software. Customers can test the output and provide feedback. According to [17] “documentation matters ... but over the years, the traditionalists made a fetish of documentation. It became the prime goal of the document-driven lifecycle” [17]. This obvious on working software over comprehensive documentation. In addition, customer involvement matters in agile methods since the customer plays a significant role. The customer is considered a team member and he/she has several tasks during the development process such as testing the output and providing feedback within the project development. Because of its importance, customer involvement has been considered on customer collaboration over contract negotiation. The last point in agile manifesto is about accepting requirements changing even later in the process. This makes agile an adaptive and more popular method, which is good in a business environment that consistently encounters change. Most traditional methods ask for well defined requirements in the beginning to start the developing process. Agile methods, on the other hand, allow projects to be developed
with no need to have well defined requirements. Some researchers believe that it is necessary to follow a certain plan while welcoming requirements change. As [17] stated “Over the years, we have learned two contradictory lessons: (1) Customers and users do not always know what they want at the outset of a software project, and we must be open to change during project execution and (2) Requirement change was one of the most common causes of software project failure” [17].

Moreover, there are twelve principles that involve the agile manifesto. These principles are as follow [18]:

1. Customer satisfaction by rapid delivery of useful software.
2. Welcome changing requirements, even late in development.
3. Working software is delivered frequently (weeks rather than months).
4. Working software is the principal measure of progress.
5. Sustainable development, able to maintain a constant pace.
6. Close, daily co-operation between business people and developers.
7. Face-to-face conversation is the best form of communication (co-location).
8. Projects are built around motivated individuals, who should be trusted.
9. Continuous attention to technical excellence and good design.
10. Simplicity-The art of maximizing the amount of work not done- is essential.
11. Self-organization teams.
12. Regular adaptation to changing circumstances.

Furthermore, being agile means rapid response to various activities such as “deliver quickly, change quickly, and change often” [17]. There are different agile techniques,
and each technique has various practices; however, all agile techniques are similar in main characteristics. For example, “including iterative development and a focus on interaction, communication, and the reduction of resource-intensive intermediate artifacts” [17]. There are some factors that help to achieve this such as working in one location, allowing the development team to communicate with each other, and make effective decisions within a short period of time. Also, dividing a project into small iterations helps the team members to quickly respond to changing requirements. In addition, “reducing intermediate artifacts that do not add value to the final deliverable means more resources can be devoted to the development of the product itself and it can be completed sooner” [17].

The following methods are known as agile development methods [17]:

1. Agile Unified Process (AUP)
2. Crystal Clear.
3. Agile Modeling (AM).
5. Adaptive Software Development.
7. Scrum.
9. Lean Software Development.
2.2 Extreme Programming (XP)

In 1998 Extreme Programming (XP) was introduced by Beck [19]. Extreme programming is one of the most successful agile methods, and it provides valuable software to the customer by following twelve practices. These practices help all team members, developers, customers, managers, and coaches to work together as a collaborative team. Also, extreme programming consists of five main values, which are as follows:

- **Simplicity:**
  Choose the simplest solution to the problem as Beck stated [19]. Simplicity encourages the development team to do only today’s work by following the “YAGNI” (You Are Not Gonna Need It) concept. As Beck states, “Simplicity is not easy. It is the hardest thing in the world not to look toward the things you’ll need to implement tomorrow and next week and next month. But compulsively thinking ahead is listening to the fear of the exponential cost of change curve” [19]. This value enhances communication between team members, and welcomes requirement changing even late in the process.

- **Feedback:**
  Feedback is a significant value in extreme programming, and might be accomplished in different ways. Unit tests can provide feedback about the system status. Also, using the acceptance test that is written with the system requirements, the customer can provide feedback. Also, “feedback works at the scale of weeks and months. The customers and testers write functional tests for all the stories (think “simplified use cases”) implemented by the system. They have concrete feedback about the current state of their system. The customers review the schedule every two or three weeks to see if the team’s overall velocity matches the plan, and to adjust the plan” [19]. As well, the entire team can give feedback when the customer adds or removes changes.
• Respect:
  In extreme programming, everyone’s effort is appreciated, and the member’s work has value. Thus, there is respect for others work, and self-respect for what he/she has also done. Also, changing should be made collaborative with other team members, which means collective decision.

• Communication:
  Communication is an important value in extreme programming. It occurs during the entire development cycle in various ways. Customers need to communicate with the developers to determine the user stories in the planning meeting. Also, developers communicate with each other to assign tasks to the programmers, and estimate the effort and cost for each user story. According to Beck “XP aims to keep the right communications flowing by employing many practices that can’t be done without communicating. They are practices that make short-term sense, like unit testing, pair programming, and task estimation. The effect of testing, pairing, and estimating is that programmers and customers and managers have to communicate” [19]. In addition, caches interact with developers to observe following extreme practices.

• Courage:
  Courage is an important value in several extreme programming practices. For example, making the design as simple as possible, in this case developers need courage to do simple design. Also, “another courageous move is throwing code away. You know how sometimes you work all day on something, and it isn’t going very well, and the machine crashes? And how the next morning you come in and in half an hour reconstruct all of the previous day’s work but clean and simple this time? Use this. If the end of the day is coming and the code is a little out of control,
toss it. Maybe save the test cases, if you like the interface you’ve designed, but maybe not. Maybe just start over from scratch” [19].

**Extreme Programming Principles:**

Extreme Programming principles are derived from the previous values. These principles provide guidelines for extreme programming process [19]. Depending on only extreme programming values will not assist the development team in using XP practices. Knowing the significance of each principle aids to select among alternatives, by selecting an alternative that comes upon the principles more than the other. These principles are divided into two groups as stated by Beck [19]. The fundamental principles are as follows [19]:

- Rapid feedback.
- Assume simplicity.
- Incremental change.
- Embracing change.
- Quality work.

The first principle is rapid feedback, which describes the importance of the respond time. After performing an action, there is feedback based on that action; therefore, reducing the time between the action and its feedback is critical. According to Beck, “one of the principles is to get feedback, interpret it, and put what is learned back into the system as quickly as possible. The business learns how the system can best contribute, and feeds back that learning in days or weeks instead of months or years” [19].

Assume simplicity is the second principle in the fundamental principles group, and this principle is hard for developers. Determining simplicity in solving a problem is not an easy thing; however, it benefits the developers to have more resources that can
be applied to different problems. In this point Beck states that “we are traditionally
told to plan for the future, to design for reuse. Instead, XP says to do a good job
(tests, refactoring, communication) of solving today’s job today and trust your ability
to add complexity in the future where you need it” [19].

The third principle in the list is incremental change. Extreme programming en-
courages small frequent changes instead of big changes. This helps the development
team solve problems with small sets of changes. In addition, “You’ll find incremental
change applied many ways in XP. The design changes a little at a time. The plan
changes a little at a time. The team changes a little at a time. Even the adoption of
XP must be taken in little steps” [19].

Embracing change: Is a principle that shows the way of solving your major prob-
lem by introducing all the possible options.

The last principle in the first group is quality work. Everyone in the development
team has a desire of doing good work. Quality of work is a significant feature in any
software project and; therefore, must always be high. If the quality of work is not
high, that means someone is not doing good work or someone does not like what they
do.

The second group of extreme programming principles is considered less central
principles. These principles are as follows [19]:

- Teach Learning.
- Small initial investment.
- Play to win.
- Concrete experiments.
- Open, honest communication.
- Work with people’s instincts, not against them.
• Accepted responsibility.

• Local adaptation.

• Travel light.

• Honest measurement.

In teaching learning principle, the concentration is on teaching ways in order to learn technical details about how much should be done. For example, how much design, testing, refactoring... etc should be done.

Small initial investment is about managing project resources, and each phase in the project reflects its needed resources. This is significant in avoiding disaster either by having too many resources or less than the required resources. Typically, each project starts with fewer resources and increases to avoid such a problem. As Beck states, “however, resources can be too tight. If you don’t have the resources to solve even one interesting problem, then the system you create is guaranteed not to be interesting. If you have someone dictating scope, dates, quality, and cost, then you are unlikely to be able to navigate to a successful conclusion” \[19\].

Play to win is another principle that encourages the development team to work to design a successful software and avoid failing. By considering this principle, software developers will focus on the various conditions that help them to achieve their success.

The concrete experiments principle deals with the process of decision making. In extreme programming, team members have to meet together, and make important decisions. These decisions are significant in the development cycle; therefore, decisions have made must be tested in order to avoid making the wrong decisions and to reduce the possible risks of making a decision.

Open, honest communication: Is important during the development process. Stating the truth is an important matter in the extreme programming process since the work is collaborative between all team members. Developers should tell the others
about their developing problems in the design or the code. Also, the developers should not be fearful of giving bad news to their customers [19].

Accepted responsibility: Team collaboration is the core of extreme programming process, and the tasks are assigned between all the team members. Every team member has responsibility of doing his/her work as assigned. When a decision is made for a specific task that needs doing, one of the team members will take responsibility and do the task.

Local adaption: According to Beck “this is an application of accepted responsibility to your development process. Adopting XP does not mean that I get to decide how you are going to develop. It means that you get to decide how to develop. I can tell you what I have found to work well. I can point out the consequences that I see from deviating. At the end of the day, however, this is your process. You have to decide on something today. You have to be aware of whether it still works tomorrow. You have to change and adapt” [19]. More details about these principles are found in [19].

2.2.1 Extreme Programming Process

The extreme programming life cycle contains five stages. These stages are exploration, planning, iterations to release, productionizing, maintenance, and death, as shown in figure 2.1

- Exploration:

In this phase, a meeting will be made between developers and customers in order to specify the wanted features in the software. The user stories will be written by the customers to show the system requirements, and the developers will ask questions to know precisely what the customers need. In addition, “at the same time the project team familiarizes themselves with the tools, technology and practices they will be using in the project. The technology to be used will be tested and the architecture possibilities for the system are explored by building
a prototype of the system” [1]. Also, the familiarity of the programmers to the technology will determine the duration of the exploration phase.

- Planning Phase:
  In the planning stage, the user stories will be prioritized by the customers based on their values. The highest user story value will be implemented first and so on. The developers will estimate the development time for each user story after breaking the user stories down into several tasks. According to [1], “the programmers first estimate how much effort each story requires and the schedule is then agreed upon. The time span of the schedule of the first release does not normally exceed two months. The planning phase itself takes a couple of days” [1].

- Iterations to release:
  Defining the iterations occurs in this stage. Each iteration will be scheduled with its user stories. Also, “the first iteration creates a system with the architecture of the whole system. This is achieved by selecting the stories that will enforce building the structure for the whole system. The customers decides the stories
to be selected for each iteration” [1]. At the end of each iteration, the customer will create the functional test and give feedback.

- **Productionizing phase:**
  This stage shows more testing to the system by checking its performance in order to release the system to the customer. Also, new changes are welcomed in this phase, so customers can add or remove features to or from the system. In addition, “during this phase, the iterations may need to be quickened from three weeks to one week. The postponed ideas and suggestions are documented for later implementation during, e.g., the maintenance phase” [1].

- **Maintenance Phase:**
  This phase is needed to support customer tasks, thus, new people might be added to the project.

- **Death Phase:**
  This is the last stage in extreme programming process, and it shows the customers satisfaction about the developed product. Moreover, “this is the time in the XP process when the necessary documentation of the system is finally written as no more changes to the architecture, design or code are made. Death may also occur if the system is not delivering the desired outcomes, or if it becomes too expensive for further development” [1].

### 2.2.2 Extreme Programming Practices

As previously mentioned, extreme programming consists of twelve practices that work together to bring valuable software to the customers. According to Abrahamsson *et al.* [1]“XP aims at enabling successful software development despite vague or constantly changing requirements in small releases and rapid feedback, customer participation, communication and coordination, continuous integration and testing, collec-
tive ownership of the code, limited documentation and pair programming are among the main characteristics of XP” [1]. These practices are described more in [19] and [1], which can be summarized as follows:

**Planing Game:**

This practice is based on a meeting between the customers and developers in order to come up with a high value product within a short period of time. This starts with writing the user stories by the customers, and the developers assist the customers by explaining the system’s needs. The developers specify the complexity and the implementation cost by estimating the user stories using story points. According to Beck, customers are responsible for:

- “Scope- How much of a problem must be solved for the system to be valuable in production? The business person is in a position to understand how much is not enough and how much is too much.

- Priority-If you could only have A or B at first, which one do you want? The business person is in a position to determine this, much more so than a programmer” [19].

Developers will be responsible for the technical issues, such as estimating the implementation time for each feature, and the decision strategies based on their technical consequences.

**Small Release:**

Small release is about producing a small version of the system, but it has benefit features. Also, “the release has to make sense as a whole-that is, you can’t implement half a feature and ship it, just to make the release cycle shorter. It is far better to plan a month or two at a time than six months or a year at a time” [19].
Metaphor:

Metaphor is used to show how the software works in a simple way. The selected metaphor should be understandable by using common vocabulary and variables that make easy to non-technical people. As Beck states “sometimes the metaphor needs a little explanation, like saying the computer should appear as a desktop, or that pension calculation is like a spreadsheet. These are all metaphors, though, because we don’t literally mean ‘the system is a spreadsheet.’ The metaphor just helps everyone on the project understand the basic elements and their relationships” [19].

Simple Design:

This reflects doing the job with the simplest design, and developers should only design for today. Simple design requires that any complexity in the code be removed right away. The simple design is the one that [19]:

- Runs all the tests.
- Has no duplicated logic.
- States every intention important to the programmers.
- Has the fewest possible classes and methods.

Test- Driven Development:

There are two types of testing in this practice. The first is unit tests, which are written by developers before the code. The second is acceptance test, which is written by the customers as a functional test. It gives feedback based on satisfaction with the system. In addition, “you don’t have to write a test for every single method you write, only production methods that could possibly break. Sometimes you just want to find out if something is possible. You go explore for half an hour. Yes, it is possible. Now you throw away your code and start over with tests” [19].
Refactoring:

Refactoring means changing the internal code without changing its external works. This might be done by making it easier to read, deleting duplication, and enhancing communication. The programmers will always look to the simplest way to add features to their works.

Pair Programming:

Two programmers work on a single machine. One is called the driver, who writes the code. The other is called the navigator, who is reviewing the code. The navigator will think in various strategies, such as is the code going to work? Or is there any way to make it more simplified, which helps in solving the problem. In addition, “pairing is dynamic. If two people pair in the morning, in the afternoon they might easily be paired with other folks. If you have responsibility for a task in an area that is unfamiliar to you, you might ask someone with recent experience to pair with you. More often, anyone on the team will do as a partner” [19].

Collective Ownership:

This practice allows any one on the team to be able to alter any part of the code at any time. No one owns the code. According to [19] “in XP, everybody takes responsibility for the whole of the system. Not everyone knows every part equally well, although everyone knows something about every part. If a pair is working and they see an opportunity to improve the code, they go ahead and improve it if it makes their life easier” [19].

Continuous Integration:

Building and integrating the system several times a day by adding a new part of the code into the base code is called continuous integration. Accepting the code depends
on passing all running tests for the code changes. In addition, “integrating one set of changes at a time works well because it is obvious who should fix a test that fails—we should, since we must have broken it, since the last pair left the tests at 100%. And if we can’t get the tests to run at 100%, we should throw away what we did and start over, since we obviously didn’t know enough to be programming that feature” [19].

40-hour week:

The development team should go home on time, so that there is no overtime work. According to Beck [19] “overtime is a symptom of a serious problem on the project. The XP rule is simple—you can’t work a second week of overtime. For one week, fine, crank and put in some extra hours. If you come in Monday and say, ’To meet our goals, we’ll have to work late again’, then you already have a problem that can’t be solved by working more hours” [19]. Also, companies should manage vacation duration issues. For example, some regions are fine with two-week vacation a year, and some need four weeks of vacation.

On-Site Customer:

In extreme programming, the customer has a significant role in the development process. The customer should be fully involved and available all the time. Beck states [19] “the big objection to this rule is that real users of the system under development are too valuable to give to the team. Managers will have to decide which is more valuable—having the software working sooner and better or having the output of one or two people. If having the system doesn’t bring more value to the business than having one more person working, perhaps the system shouldn’t be built” [19]. However, the disadvantage in this role is that it consumes customers’ time especially in the scenario when the project is cancelled. In this case, the customers have spent their time on nothing.
Coding Standards:

This reflects the code rules, and the various issues regarding to the programming phase. Rules of code should be clarified, and programming languages selected jointly. According to [19], “the standard should call for the least amount of work possible, consistent with the Once and Only Once rule (no duplicate code). The standard should emphasize communication. Finally, the standard must be adopted voluntarily by the whole team” [19].

2.3 The Analytic Network Process (ANP)

According to Saaty [20] “the Analytic Network Process (ANP) is a multi-criteria theory of measurement used to derive relative priority scales of absolute numbers from individual judgements (or from actual measurements normalized to a relative form) that also belong to a fundamental scale of absolute numbers” [20]. In addition, the ANP gives a structure in order to address a solution for a certain problem, which leads to making a decision about that problem. Also, in the ANP, dependencies among various criteria are considered, which makes it different from the Analytic Hierarchy Process (AHP) [20]. Furthermore, Saaty states [20] “in fact the ANP uses a network without the need to specify levels. As in the AHP, dominance or the relative importance of influence is a central concept. In the ANP, one provides a judgement from the fundamental scale of the AHP by answering two kinds of questions with regard to strength of dominance: 1) Given a criterion which of two elements is more dominant with respect to that criterion, 2) Which of two elements influences a third element more with respect to a criterion” [20]? In addition, in pairwise comparisons, entered values reflect the relative effect among elements with respect to a control criterion. These entered values are based on the importance of each criterion. As such, “the ANP is a useful tool for prediction and for representing a
variety of competitors with their explicitly known and implicitly assumed interactions and the relative strengths with which they wield their influence in making a decision. It is also useful in conflict resolution where there can be many opposing influences” [20]. Moreover, the network structure consists of different clusters, and these clusters contain various nodes or elements. These clusters are connected to each other based on the relative influences among the nodes. The links can either have external relative influence, which means elements in cluster X affect elements in cluster Y, or internal relative influence, which means elements in the same cluster (e.g. X) affect each other. In this case, the external relative influence is named outer-dependence, and the internal relative influence is named inner-dependence [20]. Furthermore, the network structure allows feedback models through the idea of cycle connection, and also the ANP provides different types of nodes such as source, intermediate, and sink. Again, according to Saaty [2] “a source node is an origin of paths of influence (importance) and never a destination of such paths. A sink node is a destination of paths of influence and never an origin of such paths. A full network can include source nodes; intermediate nodes that fall on paths from source nodes, lie on cycles, or fall on paths to sink nodes; and finally sink nodes” [2]. Figure 2.2 gives a general idea of the ANP structure [2].

Another issue in the ANP structure is prioritizing different alternatives in order to make an appropriate decision. This starts by making pairwise comparisons, which are based on a fundamental scale as is shown in table 2.1. Following this, “the vector of priorities is the principal eigenvector of the matrix. This vector gives the relative priority of the criteria measured on a ratio scale. That is, these priorities are unique to within multiplication by a positive constant. However, if one ensures that they sum to one they are then unique and belong to a scale of absolute numbers” [2]. In addition, “the consistency index of a matrix is given by C.I. (max n)/(n-1), where n is
the number of alternatives. The consistency ratio (C.R.) is obtained by forming the ratio of C.I. The appropriate set of numbers is shown in table 2.2, each of which is an average random consistency index computed for n 10 for very large samples. They create randomly generated reciprocal matrices using the scale 1/9, 1/8, 1/2, 1, 2, 4, 6, 8, 9 and calculate the average of their eigenvalues. This average is used to form the Random Consistency Index R.I. [2]. Moreover, the consistency ratio (C.R) should be lower than 0.10 (or 0.20), otherwise, the entered judgements need to be enhanced.

**Table 2.1:** ANP fundamental scale developed by Saaty [3]

<table>
<thead>
<tr>
<th>Scale</th>
<th>Numerical rating</th>
<th>Reciprocal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal importance</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Moderate importance of one over other</td>
<td>3</td>
<td>1/3</td>
</tr>
<tr>
<td>Very strong or demonstrated importance</td>
<td>7</td>
<td>1/7</td>
</tr>
<tr>
<td>Extreme importance</td>
<td>9</td>
<td>1/9</td>
</tr>
<tr>
<td>Intermediate values</td>
<td>2,4,6,8</td>
<td>1/2, 1/4, 1/6, 1/8</td>
</tr>
</tbody>
</table>

Furthermore, after obtaining all priorities from the pairwise comparisons, these priorities are placed in a supermatrix. According to Saaty [2] “the supermatrix rep-
Table 2.2: Random Index [2]

<table>
<thead>
<tr>
<th>Order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>R.I</td>
<td>0</td>
<td>0</td>
<td>0.52</td>
<td>0.89</td>
<td>1.11</td>
<td>1.25</td>
<td>1.35</td>
<td>1.4</td>
<td>1.45</td>
<td>1.49</td>
</tr>
</tbody>
</table>

represents the influence priority of an element on the left of the matrix on an element at the top of the matrix with respect to a particular control criterion. A supermatrix along with an example of one of its general entry matrices is shown in Figure 2.3. The component C1 in the supermatrix includes all the priority vectors derived for nodes that are parent nodes in the C1 cluster” [2].

Figure 2.3: The Super-matrix of a network [2].

2.4 Why Analytic Network Process

There are several advantages of using the Analytic Network Process (ANP) as a multi-criteria decision making tool. As stated previously, the ANP is a generalization of the Analytic Hierarchy Process (AHP); therefore, there are share benefits between the two tools over other decision making tools, and there are some advantages of the analytic network process over the analytic hierarchy process.
First of all, comparing the ANP and the AHP to other multi-criteria decision making MCDM approaches, the ANP and the AHP are less complex than other tools, which assists in enhancing “management understanding and transparency of the modelling technique” [21].

Also, ANP and AHP have the ability to merge qualitative and quantitative features into a decision. In addition, both “can be fit together with other solution approach such as optimization, and goal programming” [21]. The ANP and AHP give advantage of allowing all involved people to participate in the decision, so everyone’s voice will be heard. Moreover, “in AHP/ANP, judgement elicitations are completed using a decompositional approach, which has been shown in experimental studies to reduce decision-making errors” [21].

The following are the advantages of the ANP over AHP as shown in [22], [23], [24], and [25]:

The analytic hierarchy process requires a one way structure, which is top-to-bottom. This structure does not allow for the interdependence, top-to-bottom and bottom-to-top aspects, between elements. However, the analytic network process is non-linear in approach, which allows the possibility of interdependence among elements [23]. The power of ANP comes from having a network instead of a one direction relationship with interdependence and feedback. The ANP is more powerful than AHP in uncertainty and a dynamic decision environment [24], and [25]. In addition, “in the ANP, a rank reversal problem is appeased, thereby it is more accurate and useful (than the AHP) as a decision support instrument for intricate situations. While the ANP and the AHP are based on user supplied preferences among the factors and provide utility weights for the alternatives, they differ from each other in the number and types of pairwise comparisons, and also in the manner by which utility weights are actually computed” [21]. Moreover, the ANP allows decision makers to structure their problems closer to what happens in reality using network structure
between clusters and elements. Furthermore, order change of alternatives ranking is another problem in the hierarchy structure. According to [26] “order change is the change of alternative priorities when a new alternative is added or subtracted. This problem was minimized using the ANP suggesting that it is an appropriate decision making method for supplier selection problems among the possible potential MCDM methods” [26].

Finally, Saaty provided some ideas that support the ANP, which are as follows [2]:

- The ANP is based on the AHP.
- The ANP goes beyond the AHP by involving independence.
- The ANP handles two types of dependences: within a cluster (inner-dependence), and among different clusters (outer-dependence).
- The ANP makes it possible to represent any decision problem without concern about any order.
- The ANP ranks clusters of nodes not only elements, which is significant in real world problem.
Chapter 3

Applications of The Analytic Network Process

The analytic network process has been applied in different fields, such as industry, engineering, management, manufacturing, education, social, political, government, and so on. Also, there are various themes that ANP can be used for; for example, selection, planning and development, priority and ranking, forecasting, and evaluation.

There is a lack of research on the use of ANP in the software field. This could be because of interfering software systems with many technology areas. This section, therefore, will be divided into three portions: ANP stand alone applications in technology, ANP integrated applications in technology, and ANP applications in other fields. In addition, these applications show the different purposes of the analytic network process, such as decision, prioritization, selection, and others.

3.1 Technology Applications of the ANP

The first study is done by Ocampo and Seva [27], which highlights the use of the analytic network process in evaluating text entry methods in mobile phones with respect to preference measurements. The selected criteria are: distance between
keys, arrangement of keys, number of keys, sizes of keys, popularity and familiarity of users. This study “attempts to apply ANP in selecting text entry method by forming an evaluation model gathered from expert users and then eliciting judgements based on the decision structure” [27]. The participants are a group of expert users, who have the ability to evaluate five text entry methods in a touch keyboard smartphone. The participants reflect two issues: the usability experts, and end users with minimum five years of experience in using all text entry methods. According to [27] “the group consisted of three usability experts and seven expert users with ample experience and orientation on the use of the five text entry methods. These members were invited for a focus group discussion and were informed of their role in formulating a decision problem and eliciting judgements from this problem” [27]. The five text entry methods are: multitap (M), multitap with T9 (MT9), QWERTY (Q), the clustered handwriting style (CH), and free handwriting style (FH). Also, GT-S5570 Samsung Galaxy Mini with Android is the smartphone used in the test. The main objective of the study is to evaluate the previous methods using ANP in order to select a good text entry method. Moreover, the participants are being attentive to the different interdependences between the system comparisons; such as increasing the number of keys will affect the distance size between the keys, which also affect the arrangement of the keys. The final results show that QWERTY comes first as a good text entry method in smartphone.

Chang et al. [28] apply the ANP and AHP to evaluate digital video recorder systems. This study explains the need of the Hsinchu city government in Taiwan to enhance the monitoring of public areas. The case study shows the comparison results between the two decision models, and there is an interdependence effect in the final result. Also, the case study candidates are four digital video recorder systems that are manufactured by Firms A, B, C, and D, respectively. Each criterion are evaluated by twenty five experts in order to specify the best digital video recorder. The chosen
criteria are: functions of channel, recording, playback and display, communication and remote transmission. Data collection is done through a one hour face-to-face interview using the structured questionnaire. The system stability is tested by end users using the analytic hierarchy process and analytic network process to obtain the digital video recorder system from Firms A and B. The testing focuses on the efficiency of the CPU, the frequency of system crashes, MEM loading, and the number of successful shut-down and restarting. The final results exhibit a significant difference between the ANP and AHP models. Based on the ANP the best digital video recorder system is A; however, the AHP result shows that B is the best digital video recorder system. According to [28] “obviously, this comparative analysis in systems stable test negatively affects the result of AHP model, since the ANP decision model considers the interdependencies among selection criteria that exert additional effect on the model. AHP suffers from the limitation of only being applicable to straightforward hierarchical structures, while ANP has powerful capacity to solve current construction management issues that involve more complex decision problems” [28].

Mulebeke and Zheng [29] apply the analytic network process to select the suitable software to apply in the product development process (PDP). The objective of this application is to determine the wanted product factors for a new product. As Mulebeke and Zheng [29] state, “we considered a situation where an enterprise was interested in manufacturing pens and was interested in appropriate software to accomplish this task for their product development process” [29]. They also provide technical specifications of the product. These technical specifications include shape, user friendly, customer requirements, security, and global solution. In this study, the selected criteria are: Performance, Usability, Data and Data file support. In addition, the alternatives are: “Pro/Engineer (Pro/E) by Parametric technology corporation (PTC), Solid Edge (Sol/E) by Unigraphics solutions (UGS) and AutoCAD (ACAD) by Autodesk, from the Computer aided design (CAD) category, MasterCAM (MC)
by MasterCAM Inc., and OneCNCXR (OCNC) by One CNC, from the computer aided manufacturing (CAM) category, ANSYS by Ansys Inc., and Moldflow (Mflow) by Moldflow Inc., from the computer aided engineering category (CAE) and finally Windchill (WC) by PTC, Teamcenter (TC) by UGS, and SMARTEAM (ST) by IBM, from the product data management/ Product lifecycle management (PDM/PLM) category” [29]. The final result shows that Windchill is the best alternative in this study regarding to the goal of the study. The authors state that the ANP assists to select the appropriate alternative within an influence of multi-criteria analysis features.

Lin et al. [30] apply the analytic network process in order to optimize an enterprise resource planning (ERP) system in the electronics industry. This results in selecting the best supplier from four candidates. The alternatives are four suppliers, and the selected criteria are: price, quality, service, delivery, and trust. Also, there is a sub-criteria level that contains material, accuracy, assembly, reliability, communication, management, respond speed, credibility, capability, transportation, and so on. Lin et al. state [30] “the ERP model for purchase integrates three kinds of methodology which are ANP, TOPSIS and LP. The purchase model is applied in a large-scale high-tech firm, Asus Tech., in Taiwan; the firm is dedicated to motherboard production and has already become the largest motherboard supplier worldwide. To assemble a motherboard, hundreds of parts need to be sourced and purchased. The example used in the research is a PC board which is the main component of a motherboard” [30]. Based on the authors’ findings, the ANP helps in selecting the appropriate PC board supplier, which benefits the company in terms of saving human cost and time. In this study, the selected supplier is alternative A, which saves more cost, and increases the quality.

Erdogmus et al. [31] evaluate high-tech candidates using the ANP with rating benefits (B), opportunities (O), costs (C), and risk (R). This study reflects the importance of adapting a new technology with considering the benefits, costs, and the risk of the
adaption. According to Erdogmus et al. [31] “the major motivation towards the problem studied in this research is the renewal of a transaction processing system (TPS) for enhanced efficiency and operation quality in Osmangazi University Consumption Cooperative (OTK), with 1350 members and 294 contracted stores, founded by the personnel of the university. In search of increased competitiveness, the management faced a decision-making problem either to maintain the current financial TPS or to upgrade the system to a new technological POS (Point-Of-Sale) based TPS. Then, the management collected the alternative proposals from four vendors with each having different features of POS-based systems” [31]. The authors provide the selected criteria for each factor (e.g. benefits, opportunities, costs, risk). The chosen benefits criteria are: ease of transaction, ease of control, image, a reduction in transaction cost, ease of solving problems, and ease of usage. The selected criteria for opportunity are: increasing number of members, competition power of coop, joining new stores, more shopping, and benefiting from sales and campaigns. The cost criteria are: machinery cost, card/printing cost, commission of the card vendor, and labour cost. Finally, the risk criteria include withdrawal of a store, risks associated with loosing cards, and vendor’s credibility. The alternatives are five transaction-processing systems. Four are new systems, and the fifth one is the current transaction system. 68 members and 15 store managers are involved to participate in this research. The final results of this study show that transaction system D is the best system to replace the current one. Based on the research, the OKT management accepted to replace the current transaction system with system D as this study recommends.

Lee et al. [32] apply the analytic network process to select the technology acquisition mode. Thus, the study aims to choose the best solution for acquiring the wanted technology among various strategies or modes. Also, Lee et al. state [32] “a number of technology acquisition modes are available, such as acquisition, merger, licensing, joint venture, joint R&D, R&D contract, alliance, consortium, outsourcing,
in-house R&D. Since too many alternatives make the ANP procedure extremely complex and time-consuming, three broad categories of the technology acquisition modes have been defined as the alternatives of the ANP model: Make, Cooperate, and Buy. Make means in-house R&D, and Cooperate includes various forms of cooperation with other firms with or without equity involvement such as joint venture, joint R&D and alliance. Buy constitutes a form of R&D contract, acquisition, licensing, and outsourcing” [32]. In addition, the selected criteria are: capability, strategy, technology, market, and environment. To specify the above general procedure, a case study was conducted that shows a software company located in Seoul, Korea, wanting to enter a new technology field. This requires the best selection of acquiring strategy, which needs to be applied by the ANP. Lee et al. exhibit [32] “although the main product has been middle-ware solutions, the company has a plan to enter the small and medium enterprise (SME) enterprise resource planning (ERP) market by developing their own ERP package. The technologies or products required for developing the designed ERP package are as follows: (1) AJAX, (2) OR mapping, (3) Aspect oriented programming (AOP), (4) Role based access control (RBAC), and (5) Group ware. The company already possesses the high level of AJAX and group ware technologies, and OR mapping and AOP can be obtained as a freeware. The problem to be faced is how to acquire the RBAC technology” [32]. After completing the pairwise matrices, the final results recommend acquiring the RBAC alternative through in-house R&D.

Xu and Guo [33] apply the analytic network process in order to optimize cloud computing services composition. The model aims to achieve high user satisfaction in order to consider success. As Xu and Guo state [33] “selection of optimal service composition using ANP is an application process of ANP from receiving the requests from service requesters to accomplish the selection of optimal service composition with considering decision-maker preference. The optimization selection process includes not only the ANP evaluation process, but also the establishment of metrics
for measuring service composition quality, optimizing ANP alternative set according to the requirements of users, assignment to alternatives using interactive simulation and other important processes” [33]. Service composition quality is being considered in order to measure the superiority of service composition. Also, three service composition features are considered, which are cost, hardware resource constraints and service composition quality. Three criteria are selected in this model, which are goal, control hierarchy, and network elements. Each criterion contains several indicators as sub-criteria. For example, CPU resource constraints, memory resource constraints, and I/O resource constraints are the indicators under hardware resource constraints criterion. Service cost and service level agreement (SLA) violation are the indicators under service composition cost criterion. The final application for the ANP shows that optimal selection is service composition 3, and this comes from the high weight of service composition cost indicators.

Wang and Liu [34] evaluate cloud service architecture using the analytic network process. Non-functional attributes of cloud computing system are used to measure the architecture. The study aims to rank the quality of service (QoS) of cloud service architecture using ANP. According to Wang and Liu [34] “this paper creates the enterprise cloud service architecture design model based on the Department of Defense Architecture Framework Version 2.0 (DoDAF V2.0), the Open Group Architecture Framework (TOGAF) and the Cloud Computing Modeling Notation (CCMN)” [34]. An architecture that can be seen as a filter mode is the optimal architecture, which should be achieved by applying the ANP. In this case study, five systems are evaluated as alternatives, which are: the visitor management system, the catering management system, the enterprise information portal, the parking management system, and the facility management system. There are three main criteria clusters, which are: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). Each criterion consists of several items; for example, network delay time, ap-
Application system response time, application system throughput rate, energy consumption, and host utilization rate are the elements under the infrastructure as a service cluster. Also, extensibility, service request error rate, service request throughput, service delivery stability, development environment convenience, automation deployment efficiency are the sub-criteria in platform as a service cluster. Software as a service cluster contains service response time, service cost, service availability, service reliability, and service credibility. The ANP results rank the enterprise information portal as the optimal cloud service architecture, while the catering management system gets the lowest rank.

Babu et al. [35] introduce the analytic network process to select the appropriate architecture style in order to optimize software architecture. In this study, the presented algorithm (ANP) contains both qualitative and quantitative approaches. For the qualitative approach, the architect problem is determined, based on the customer satisfaction. For example, selecting the best architecture style from different available styles. Another qualitative issue is the suitability of the ANP to solve the decision problem. Also, “decompose the unstructured problem of selecting the architecture style in to a set of manageable and measurable levels. The topmost level is the prioritization of styles, while the lowest level is usually the scenario or architecture styles” [35]. The last qualitative point involves the choosing participants, who are going to evaluate different architecture styles. The quantitative issues include setting up quantitative data collection methods, such as a quantitative questionnaire to collect priorities from participants. As well, pairwise comparison matrices are used in order to test the relative importance among elements, the inconsistency is measured and the eigenvectors are placed for each pairwise comparison. The participants are asked to evaluate four alternatives (e.g. Architecture styles), and four criteria. The selected criteria are performance, flexibility, reusability, and maintenance. The four alternatives are pipes-and-filters, layered, blackboard, and abstract data type.
final estimated weights show that pipes-and-filters is the best architecture style, so it should be selected. The authors emphasize “the results obtained from this approach (e.g. ANP) are better than any other approaches followed to select architecture style such are decision support systems, Fuzzy approach, AHP, AHP-GP” [35].

Nazir et al. [35] apply the analytic network process (ANP) to select software component based on quality criteria. In this case study, the goal is to select the appropriate software component, which provides all the wanted functionalities. Selecting the suitable component is significant to ensure software success. There are four candidates software components (C1, C2, C3, and C4) as alternatives, and five criteria. These criteria are effectiveness, efficiency, satisfaction, safety, and usability. Also, each criterion has sub-criteria; for example, likability, pleasure, comfort, and trust are sub-criteria under satisfaction criterion. In addition, economic damage risk, health and safety risk, and environmental harm risk are the sub-criteria under safety criterion. Usability criterion has the follow sub-criteria: Learnability, flexibility, accessibility, and context conformity. Moreover, individual judgements are derived from a set of 15 experts. The final results show that component 1 (C1) is the best software component selection in this case study. Selecting component 1 has several advantages, such as reducing cost, increasing productivity, and enhancing overall quality. Also, according to Nazir et al. [35] “the results of the proposed method clearly show that the method is quite beneficial and favourable in decision making regarding the most suitable software component selection. Hence, it is concluded that ANP is one of the best choices for software component selection” [35].

Dimitrova [37] introduces a framework of Enterprise Resource Planning (ERP) systems implementation success using the analytic network process. According to Dimitrova [37], “we propose a computer-based methodology for evaluating and selecting the alternative by ERP systems implementation project to improvement the success and the total outcome of the project. The objective here is to develop a
framework to analyze success factors specific to one ERP systems implementation project and to select alternatives and success factors-based activities to improve the success. From this it allows that the total outcome of the project can be improved. The introduced framework contains five stages, which are specifying the related criteria and alternatives, structuring the problem, prioritizing the selected criteria, evaluating the inner-dependence among components, and the final stage is selecting the best alternative. To achieve this goal, three alternatives are selected, which are activity in field motivation (A1), activity in field application of methods and tools in the managing of the projects (A2), and activity in field communication (A3). In addition, three criteria are chosen, which are quality (Q), cost (C), time (T). The framework has several success factors such as application of promoters, top management support, corporate culture, ability to change, knowledge of the market, skills, competence and qualification, cooperation of vendors and consultants, and cooperation of technical and business experts. After following the ANP steps, the final results indicate that A2 is the best alternative, so it should be selected.

Godse et al. rank web services based on the analytic network process. The article shows that selection web service is important regarding the various non-functional requirements that each service provides. Since this issue depends on the multi-criteria decision making approach, the analytic network process is used. According to Godse et al. “this paper proposes Analytical Network Process (ANP) approach to solve the problem of assigning weights to features considering interdependence between them, thus providing a much-needed quantitative basis for web service selection.” The authors determine three criteria, which are runtime, security, and configuration management. Each criterion contains several parameters or elements; for example, runtime criterion consists of availability (AL), reliability (RL), response time (RT), and throughput (TP). Also, security criterion contains authentication (AE), authorization (AO), confidentiality (CD), and non-repudiation (NR). The sub-criteria un-
der configuration management are interoperability (IO), regulatory (RL), stability (SB), and supported standard (SS). The article aims to rank three web services for online share trading. These three services have the same functional requirements, but they have various non-functional requirements, which affect the user satisfaction. The three services (e.g. service 1, service 2, service 3) are selected with respect to the requirement from user opinion, and desirability of the values regarding to the parameter. One example of evaluating the elements is that minimum responding time is preferable to get better service; therefore, the service that has lower responding time will get a high score. The final ranking of the services shows that service 2 is selected because it has the highest rank.

Sun and Wang [39] use the analytic network process to study factors that affect customers' online shopping behaviour. The authors introduce three main clusters to be studied. These clusters are website professionalism, retailers, and consumer factors. Each cluster consists of several nodes such as production introduction, web navigation design, ease of use of settlement, and the security of network transaction; these are the nodes in the website professionalism cluster. Retailers cluster has the following nodes: retailer's reputation, product type, and service quality. Consumer factors cluster contains education level, network and computer operations experience, network consumption consciousness, and household income. In the article, the authors emphasize the benefits of applying the ANP tool in their study, and the ANP helps the researchers to find out the weight for each factor in their model. By determining the importance for each factor, improvement for the online transaction can be made by considering the important factors. The final results of the study show that the security of network transactions has the highest weight.

He [40] applies the ANP to evaluate online customer trust in business to consumer (B2C) e-commerce. The study exhibits a comparison between three business to consumer companies with respect to several criteria and indicators. The study aims to
increase customer’s trust in order to increase their business. In this article, the model consists of five clusters. First, company cluster contains two sub-elements, which are scale of the company, and brand of the company. Second, website cluster has website quality, reputation of website, relationship marketing, and website security. Disposition to trust, and perception are under the third, consumer cluster. Fourth, environment cluster includes law and technique, financial intermediary, and logistics supplier. Finally, the alternative cluster has three alternatives (e.g. business to consumer companies), which are 360buy, dangdang, and m18. After deriving all the pairwise comparisons, the author states that the three indicators that have highest impact on trust are company brand, law and technique. Also, the 360buy company has more customer trust than the other alternatives. The author concludes with that the ANP helps to reflect the reality of customer trust in business to consumer e-commerce.

Nazir et al. [41] evaluates software component security using the analytic network process. According to Nazir et al. [41] the use of the analytic network process is to “evaluate security of software components. The technique incorporates attributes of ISO/IEC 27002 standard. The goal is to evaluate available software components in order to find the most secure component to be used for component based software development” [41]. The case study involves several experts to specify a search component, and the study shows three selected searching components which are SC1, SC2, and SC3. Each search component will be evaluated with respect to its security. The one that has highest security is the winner. In addition, eleven criteria are selected for the ANP model. These criteria are 1) security policy (SP), 2) organization of information security (OIS), 3) asset management (AM), 4) human resources security (HRS), 5) physical and environmental security (PES), 6) communications and operations management (COM), 7) access control (AC), 8) information systems acquisition development and maintenance (ISADM), 9) information security incident
management (ISIM), 10) business continuity management (BCM) and 11) compliance (C). The final results show searching component 1, SC1, has the highest quality secure. The article concludes that the ANP is a powerful and valuable tool to evaluate software component security.

Pandey and Agrawal [42] estimate the quality of software components using the analytical network process. This article aims to prioritize the quality characteristics and sub-characteristics, and estimate software quality value. Also, Pandey and Agrawal state, [42] “we have decided to take into account our institute biometric system, which is designed and developed in the institute and used for employee attendance management. Basically it is the component used in the employee management system of our institute. The goal of the proposed model is to prioritize the quality characteristics while designing the biometric system”. The authors select 20 participants to complete their survey on software in biometric based projects. In this study, the characteristics are functionality, reliability, usability, efficiency, maintainability, portability. In addition, the sub-characteristics are suitability, accurateness, interoperability, compliance, security, maturity, fault tolerance, recoverability, understandability, learnability, operability, time behaviour, resource behaviour, testability, stability, changeability, adaptability, install-ability, replaceability. After applying the analytic network process, the final results show that functionality is the most important quality characteristic software component.

Buitrago and Lesmes [43] select new markets to export software services using the analytic network process. The study investigates decision making in small and medium enterprises (SMEs) in Colombia. The old way shows that small and medium enterprises register in export programs, and based on their results, companies select their foreign markets. One of these programs is called a Trade Promotion Organization (TPO). This article introduces the ANP as a more powerful tool to select the new markets. At the end of this study, the authors compare the results of the two tools
(e.g. ANP and TPO). The model of this article shows that seven clusters represent
different criteria. These clusters are economy, political stability, foreign trade, logistic,
trade policy, market, and company’s experience. Each cluster contains several nodes,
for example, economy cluster has inflation and devaluation nodes. Political stability
cluster includes country risk node. Foreign trade cluster has export from Colombia
and main suppliers. Logistics criterion contains calling cost per minute, flight dura-
tion per hour, and ticket cost. Trade policy cluster involves trade agreements, taxes,
and intellectual property. Market criterion consists of target population, public price,
pirate price, and main competitors. The final cluster is company’s experience, and
it has exploratory exports and quality certificates nodes. There are various countries
selected as alternatives in this study. These alternatives are Costa Rica, Ecuador, Ar-
gentina, Chile, USA, Venezuela, Peru, Guatemala, Brazil and Mexico. These criteria
and alternatives are compared with each other by experts in foreign trade. Using su-
perdecision software, the authors find that USA is the target country, Venezuela is the
alternative country, and Guatemala is the contingent country. Moreover, both TPO
and ANP provide similar results to this case study, however, the authors recommend
the analytic network process regarding that [43] “it has a structured methodology to
assign weights that allows evaluating the consistency of the weights allocated to the
selected criteria. Besides, we also preferred it because allows analyzing sensitivity
to changes in the priority weights. The results of the ANP model generate more
confidence than the results of the TPO model because the subjectivity is reduced”
[43].

Ayag [44] evaluates simulation software alternatives in order to reach to the appro-
priate alternative. The author selects four criteria, which are hardware and software
consideration, modelling capabilities, simulation capabilities, and input/output. Each
criterion includes several nodes, for example, hardware and software consideration has
pedigree, coding aspects, software compatibility, user support, financial, and techni-
cal features. Modelling cluster involves general features and modelling assistance. Simulation capabilities have visual aspects, efficiency, testability, experimentation facilities, and statistical facilities. Input/output cluster contains input/output capabilities, analysis capabilities. The author specifies three alternatives, which are Arena, Flexsim, and Promodel. After using the super decision software to calculate the enter judgements, the final result emphasizes that Arena has the highest weight among the alternatives.

Raisinghani et al. [45] use the analytic network process to select the optimal e-business strategies. The study investigates the interdependence between four e-commerce alternatives, which are virtual information space (VIS), virtual transaction space (VTS), virtual communication space (VCS), and virtual distributed space (VDS). Also, the authors identify two general business strategies, which are cost leadership and differentiation. Each strategy consists of several attributes that have inter-influence among each other. For instance, cost leadership strategy has process, backward vertical integration, automation with efficient scale facilities, and minimization of SG&A expenses. On the other hand, differentiation strategy has innovation-based differentiation, marketing based differentiation, and response time-based differentiation. The data was collected through a live web-based survey, and through hard copy questionnaires. The final analysis shows that the virtual information space alternative is the optimal e-business strategy.

Viglas et al. [46] apply the ANP in order to select the suitable information system. As Viglas et al. state [46], “the proposed model provides a simple, flexible and easy to use approach that can be applied by organizations to support their investment decisions. The proposed approach is presented through a case study for selecting a Quality Management Information System for a large Greek retailer” [46]. The process of selecting this type of information system is affected by various factors such as investment cost, investment profit, and return on investment. Performance is
also a significant measure in choosing information system. For the analytic network process, the authors identify four clusters as criteria, which are cost, functionality, supplier, and technology. Each cluster has sub-criteria; for example, cost cluster includes implementation cost, license cost, and maintenance cost. Functionality cluster has email notification, flexibility, product audit, reports, store audit, supplier audit, and security. Supplier criterion contains company profile, implementation time, and international solution. The technology cluster has database, database connectivity, infrastructure, migration tools, reporting tools, and web application. System A, system B, and system C are selected as alternatives to be weighted. After following the analytic network process steps, the findings indicate that system A is the best quality management system that should be selected.

Feili et al. [47] introduce the analytical network process in order to select the best enterprise resource planning (ERP) system. The article presents a case study based on petrochemical industry field. The objective of this work is to help the organization to select the best enterprise resource planning system. The chosen system might help organizations in various activities such as planning, accounting, distribution, marketing, sales, production, project management, and human resource management. The ANP model contains seven criteria, which are general feature, implementation cost, price, vendor, software capabilities, implementation and project management, and software quality. Each criterion has various sub-elements, for example, flexibility, maintenance, and hardware are under general feature criterion. Implementation cost cluster has consultation costs, infrastructure costs, and implementation time. Price cluster includes the degree of localization, vendor estimate, having full operations, and having complete modules. Vender criterion has seller’s fame, technical and financial abilities of the vendor, experience and skills of the key persons assigned to the project, timely support services, experience of the vendor, and market share. Also, learning ability, efficiency, appropriateness of the modules, user friendliness, open
source, open system interface, comprehensive performance, future promotion and update, localization abilities, and online documentation and subject area help are the elements in the software capabilities cluster. Implementation and project management criterion includes implementation methodology, project control program, and project management. The final criterion is software quality, and its sub-criteria are testability, performance level, accessibility, performance, learning abilities, efficiency, memorization ability, error avoidance, dealing with errors, integration ability, and security. In addition, the article shows three enterprise resource planning systems as alternatives. These systems are selected based on their marketing ranking. The three alternatives are namely system A, system B, and system C. The authors conclude with that system B gets the highest weight so should be selected.

Lee and Kim [48] select the suitable information system project using the analytic network process. The authors focus on the major values, such as cost saving and benefits to organizations, that can be provided by a good information system. Also, expert interview is used to collect data, and enter user judgements. The study shows four main criteria, which are, increased accuracy in clerical operations (AC), information processing efficiency (E), promotion of organizational learning (OL), and cost of implementation (CI). Moreover, there are six information system projects selected as alternatives namely P1, P2, P3, P4, P5, and P6. The final analysis indicates that information system project 6 is the best one among the alternatives. The article concludes with [48], “in project selection, it is very important to consider the interdependent relationship among projects or criteria because of the characteristics of interdependence that exists in real problems. In addition, the cost of difficulty in data gathering for modelling is not so critical than the risk in selecting the wrong project without considering the interdependencies” [48]. In other words, ANP has the power in solving such problems regarding its ability to structure the problem in a network while considering the interdependences among network components.
<table>
<thead>
<tr>
<th>Article Title</th>
<th>ANP Purpose</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using analytic network process for evaluating mobile text entry methods</td>
<td>Evaluation</td>
<td>Ocampo and Seva[27]</td>
</tr>
<tr>
<td>Evaluating digital video recorder systems using analytic hierarchy and analytic network processes</td>
<td>Evaluation</td>
<td>Chang et al. [28]</td>
</tr>
<tr>
<td>An ERP model for supplier selection in electronics industry</td>
<td>Selection</td>
<td>Lin et al. [30]</td>
</tr>
<tr>
<td>Evaluating high-tech alternatives by using analytic network process with BOCR and multiactors</td>
<td>Evaluation</td>
<td>Erdogmus et al. [31]</td>
</tr>
<tr>
<td>Selection of technology acquisition mode using the analytic network process</td>
<td>Selection</td>
<td>Lee et al. [32]</td>
</tr>
<tr>
<td>Optimization Strategy of Cloud Computing Service Composition Research Based On ANP</td>
<td>Selection</td>
<td>Xu and Guo [33]</td>
</tr>
<tr>
<td>QoS EVALUATION OF CLOUD SERVICE ARCHITECTURE BASED ON ANP</td>
<td>Evaluation</td>
<td>Wang and Lin [34]</td>
</tr>
<tr>
<td>Software Component Selection Based on Quality Criteria Using the Analytic Network Process</td>
<td>Selection</td>
<td>Nazir et al. [36]</td>
</tr>
<tr>
<td>Application of the Analytic Network Process (ANP) in a framework of ERP systems implementation success</td>
<td>Selection</td>
<td>V.A Dimitrova [37]</td>
</tr>
<tr>
<td>Web Service Selection based on Analytical Network Process Approach</td>
<td>Selection</td>
<td>Godse et al. [38]</td>
</tr>
<tr>
<td>A Study of Factors Influencing Customers' Online Shopping Behavior based on ANP</td>
<td>Prioritization</td>
<td>Sun and Wang [39]</td>
</tr>
<tr>
<td>An Analytic Network Process Model of Trust in B2C E-commerce</td>
<td>Prioritization</td>
<td>He, Juan [40]</td>
</tr>
<tr>
<td>Evaluating security of software components using analytic network process</td>
<td>Evaluation</td>
<td>Nazir et al. [41]</td>
</tr>
<tr>
<td>Analytical Network Process based Model to Estimate the Quality of Software Components</td>
<td>Evaluation</td>
<td>Pandey and Agrawal [42]</td>
</tr>
<tr>
<td>Application of the analytic network process (ANP) to select new foreign markets to export software services: study of colombian firms</td>
<td>Selection</td>
<td>Buitrago and Lesmes [43]</td>
</tr>
<tr>
<td>Evaluating Simulation Software Alternatives through ANP</td>
<td>Evaluation</td>
<td>Ayag, Zeki [44]</td>
</tr>
<tr>
<td>Strategic e-Business Decision Analysis Using the Analytic Network Process</td>
<td>Selection</td>
<td>Raisinghani et al. [45]</td>
</tr>
<tr>
<td>An Integrated Approach for Selecting Information Systems: A Case Study</td>
<td>Selection</td>
<td>Viglas et al. [46]</td>
</tr>
<tr>
<td>A multi-stage approach to enterprise resource planning system selection</td>
<td>Selection</td>
<td>Feili et al. [47]</td>
</tr>
<tr>
<td>Using analytic network process and goal programming for interdependent information system project selection</td>
<td>Selection</td>
<td>Lee and Kim [48]</td>
</tr>
<tr>
<td>An ERP software selection process with using artificial neural network based on analytic network process approach</td>
<td>Selection</td>
<td>Yazgan et al. [49]</td>
</tr>
<tr>
<td>Analytic network process for pattern classification problems using genetic algorithms</td>
<td>Other</td>
<td>Yi-Chung Hu [50]</td>
</tr>
<tr>
<td>A customizable web service selection framework using MCDM approaches</td>
<td>Selection</td>
<td>Raed Karim [51]</td>
</tr>
</tbody>
</table>

Table 3.1: Summary of the ANP Applications in Technology with their purposes
3.2 ANP Integration in Technology Applications

The analytic network process is a tool that is able to integrate with one or more tools in order to achieve the appropriate decision. Therefore, in this section, I will review some works that integrated the analytical network process with different tools.

Gurbuz et al. [52] integrate the analytic network process with two different tools, Choquet Integral (CI) and measuring attractiveness by a categorical based evaluation technique (MACBETH). This is done in order to select the most appropriate enterprise resource-planning project according to an organization’s requirements. The ANP is used to prioritize four alternatives, which are four enterprise resource-planning projects. Also, the choquet integral and the process of measuring attractiveness by a categorical based evaluation technique are used to specify disjunctive or conjunctive conduct between the selected criteria. In this case study, three main criteria are chosen, and they are vendor related criteria (VRC), customer related criteria (CRC), and software related criteria (SRC). Each criterion has sub-criteria. For example, vendor related criteria have support and services, vision, market position, domain knowledge, reputation, and methodology of software. Customer related criteria includes ease of customization, better fit with organizational structure, fit with parent/allied organizational system, and cross module integration. Software related criteria cluster consists of functionality, technical aspects, cost, system reliability, compatibility, and implementation time. After obtaining individual judgements by a decision maker, the final ranking for A1, A2, A3, and A4 is 0.686, 0.642, 0.478, and 0.470 respectively. In other words, alternative 1 is the most appropriate ERP project, so it should be selected.

Shih et al. [53] apply the analytic network process integrated with decision making trial and evaluation laboratory (DEMATEL) to determine the suitable system for implementation decision-making. According to Shih et al. [53] “this study applies the DEMATEL to establish the correlation between the mutual effects of the organiza-
tional perspectives, and uses ANP to determine the weights of the information system and system implementation decision making" [53]. The article specifies eight information system criteria, which are financial accounting system (FAS), management accounting system (MAS), advanced planning and scheduling (APS), supply chain management (SCM), customer relationship management (CRM), knowledge management (KM), executive information system (EIS), and decision support system (DSS). In addition, the authors identify three system implementation models as alternatives, which are in-sourcing, out-sourcing, and co-sourcing. The judgements are obtained by information system experts. The final analysis shows that the out-sourcing model has the highest weight, which means out-sourcing is the best solution.

Nazir et al. [54] integrate the ANP with fuzzy approach in order to prioritize component test case. The authors specify six criteria, which are controllability (C), observe ability (O), isolatability (I), separation of concern (S), automatibility (A), and heterogeneity (H). In addition, the article selects three component test cases as alternatives to be ranked regarding to experts’ judgements. The authors conclude with that component test case 2 is the top alternative, which should be chosen.

Shahzad et al. [55] evaluate a component based project using a fuzzy analytical network process. The aim of this study is to determine which component has more importance. According to Shahzad et al. [55] “quality components are the foundation to guarantee the quality of the whole component based software system. The proposed method incorporates the attributes of ISO/IEC 25010:2011 to help in the evaluation of component based projects” [55]. The authors identify five criteria for their model. These are effectiveness, efficiency, satisfaction, safety, and usability. Each criterion has sub-criteria. For example, satisfaction criterion has likability, pleasure, comfort, and trust. Safety criterion includes economic damage risk, health and safety risk, and environmental harm risk. Learnability, flexibility, accessibility, and context conformity are the sub-criteria in usability cluster. Three software component projects are
selected as alternatives named software component project 1 (CP1), software component project 2 (CP2), and software component project 3 (CP3). Individual opinions are obtained from a group of eight experts. The final evaluation shows the top component is software component project 3 (CP3) followed by software component project 1 and then software component project 2. Finally, the authors conclude that the fuzzy analytical network process is a powerful tool for handling problems with uncertain elements.

Etaati et al. [56] introduce the integration of fuzzy and analytical network process to evaluate ISO 9126 quality model factors. ISO 9126 is a tool that is used to assess software quality. The objective of this study is to assess three e-learning software models that are used in three Iranian universities. The study contains nine main criteria, which are functionality, reliability, usability, efficiency, maintainability, portability, learning interface, learner community, and system content personalization. Each criterion contains several sub-criteria. For example, suitability, accuracy, interoperability, and security are in the functionality cluster. The reliability cluster includes maturity, fault tolerance, and recoverability. Usability criterion has understandability, learnability, attractiveness, and operability. Efficiency consists of time behaviour and resource utilization. Maintainability cluster includes analyzability, changeability, and testability. Portability has adaptability, install-ability, and replaceability. Learning interface contains ease of use, user friendly, and ease of understanding. Learner community includes ease of discussion with students, teacher, and sharing data. Personalization has capability of controlling and recording. The three e-learning software systems selected are Iran University of Science and Technology (IUST), AmirKabir University of Technology (AUT), and Virtual University of Shiraz (SVU). The data collection process includes various students, developers, and managers of the three systems. Data is collected by survey and face-to-face interviews. The final assessment shows that Virtual University of Shiraz (SVU) is the best e-learning system based on
Ayag and ozdemir [57] apply the integration of the fuzzy approach with the analytic network process in order to solve machine tool selection (MTS) problems. The authors evaluate different kinds of machine tools that are widely used in manufacturing systems. The studied model contains three levels of criteria. The first level contains improved customer satisfaction and increased profitability. The second level includes increased productivity, higher flexibility, effective use of space, better adaptability, better precision and accuracy, increased reliability, more safety and environment, and satisfied maintenance and service. The third criteria level has spindle speed, main power, cutting feed, traverse speed, tool change time, capacity of rotary table, average set-up time for product change, machine dimensions, area for accessories, repeatability, thermal deformation, checking probe installed, bearing failure rate, reliability of drive system, reliability of computer controlled system, operator training for safety, proportion of recycling components, safety accessories, specialized training, on-time repair service, and regular maintenance. The article identifies three machine tools as alternatives named machine tool 1 (MT1), machine tool 2 (MT2), and machine tool 3 (MT3). Individual judgements are obtained from experts in the field. The end of the analysis indicates that machine tool 1 (MT1) is selected as the best alternative.

Zhou et al. [58] measure the flexibility of enterprise resource planning (ERP) system by integrating fuzzy with analytic network process and fuzzy preferences programming (FPP). Zhou et al. state, [58] “flexibility measurement is an important item for the implementation of ERP flexibility. According to the characteristics of ERP system, an index system for flexibility measurement of ERP system is presented with the interdependence and feedback relationships among criteria and/or indices being taken into account” [58]. The article describes the need for a technology company to adapt a new enterprise resource planning system in order to compete with other com-
petitors in the market. Also, the authors specify the flexibility level of an enterprise resource planning system, which is as follows: <0.40 =bad, 0.40-0.55=poor, 0.55-0.70=general, 0.70-0.85=good, and >0.85=excellent. The proposed model has two levels of criteria. The first level has the main criteria, which are function flexibility, transaction process flexibility, client flexibility, and responsiveness flexibility. The second level includes degree of structuring, adaptability, structure expansibility, kernel stability, module coupling degree, parametric design, matching degree, flexibility of configuration, component based business, business adaptability, business reconfiguration, redefinition of process documents, redefinition of input and output, redefinition of interface, online job response time, task switching speed, and accuracy. The third level of the model shows the goal, which is flexibility measurement of enterprise resource planning system. After integrating the fuzzy analytical network process and fuzzy preferences programming, the final score indicates that the new ERP system gets 0.7253, which means the enterprise resource planning system flexibility level is good.

Partani et al. [59] apply the fuzzy analytical network process integration into a strengths, weaknesses, opportunities and threats (SWOT) analysis. The objective of the study is to assist an organization to select the appropriate strategy in order to survive among other competitors. The authors apply the fuzzy analytical hierarchy process in addition to the fuzzy analytical network process in order to test the two tools. According to Partani et al. [59], “the ANP powerful instrument with fuzzy phase got to allow modelling SWOT analysis to planners organization. This research was done in 2011, in Notash Afra Co. which works in the field of installation of water and energy projects in Tehran” [59]. The model consists of four main criteria as follows: strengths, weaknesses, opportunities and threats. Each criterion has sub-criteria, for example, strengths includes technical skills of staff, no restriction in recruiting of skilled man-power, strong information system and software, the spirit of
teamwork, and good relationship with technology owners. Weaknesses criteria include being a young company, a lack of acquisition in equipment, a lack of experience in oil projects, a lack of quality control system, and a lack of equipment and proper infrastructure. Opportunities involve the country’s abundant energy resources, restrictions for foreign contractors, continued growth in domestic international demand for energy, and the weakness of the region countries. Threats include instability in the economic environment, the presence of competitors with well-known brand names, government policies in line with privatization, low labour productivity in the country, and significant market share of competitors. There are four selected strategies as alternatives, which are “SO Strategy: market development - the foreign goal market, WO Strategy: professional reinforcing of manpower and infrastructure in the area of thermal power plants, ST Strategies: development and implementing of new technologies, and WO Strategies: cooperation and strategic partnership” [59]. The final ranking shows that WO strategy is best for the mentioned company. The authors conclude that in comparing the fuzzy analytical network process (FANP) with the fuzzy analytical hierarchy process (FAHP), there is a difference between the two approaches; however, the fuzzy analytical network process has more accurate results because it considers the interdependence among the problem factors, which are more realistic.

Sugiyanto and Rochimah [60] integrate the analytic network process with decision making trial and evaluation laboratory (DEMATEL) in order to find out the weights of characteristics software quality based model ISO 9126. The characteristics of model ISO 9126 is divided into two levels. The first level contains main characteristics, and the second level their sub-characteristics. The main characteristics are functionality, reliability, usability, efficiency, maintainability, and portability. Each one has several sub-characteristics, for example, functionality has suitability, accuracy, interoperability, compliance, and security. Reliability includes maturity, fault tolerance, and recoverability. Usability contains understandability, learnability,
operability, and attractiveness. Efficiency consists of time behaviour, and resource utilization. Maintainability has analyzability, changeability, stability, and testability. Finally, portability has adaptability, install-ability, conformance, and replaceability. The authors integrate ANP with DEMATEL to measure the weight for each characteristic. The final results show that sub-characteristic fault tolerance has the highest weight and most impact in software quality, so it deserves more importance than other characteristics.

<table>
<thead>
<tr>
<th>Article Title</th>
<th>Integrated With</th>
<th>ANP Purpose</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>An Integrated Decision Support System for Selecting Software Systems</td>
<td>CI and MACBETH</td>
<td>Prioritization</td>
<td>Gurbuz et al. [52]</td>
</tr>
<tr>
<td>Applying DEMATEL-ANP for Assessing organizational information System Development Decision</td>
<td>DEMATEL</td>
<td>Evaluation</td>
<td>Shih et al. [53]</td>
</tr>
<tr>
<td>Test case prioritization for components using FANP</td>
<td>Fuzzy Set</td>
<td>Prioritization</td>
<td>Nazir et al. [54]</td>
</tr>
<tr>
<td>Software component project evaluation based on quality measure</td>
<td>Fuzzy Set</td>
<td>Evaluation</td>
<td>Shahzad et al. [55]</td>
</tr>
<tr>
<td>An intelligent approach to machine tool selection through fuzzy analytic network process</td>
<td>Fuzzy Set</td>
<td>Selection</td>
<td>Ayag and ozdemir [57]</td>
</tr>
<tr>
<td>ERP System Flexibility Measurement Based on Fuzzy Analytic Network Process</td>
<td>Fuzzy Set and Fuzzy Preferences Programming</td>
<td>Assessment</td>
<td>Zhou et al. [58]</td>
</tr>
<tr>
<td>Using Fuzzy Analytic Network Process (FANP) in a SWOT Analysis</td>
<td>Fuzzy Set</td>
<td>Selection</td>
<td>Partani et al. [59]</td>
</tr>
<tr>
<td>Integration of DEMATEL and ANP Methods for Calculate The Weight of Characteristics Software Quality B based Model I S O 9 1 26</td>
<td>DEMATEL</td>
<td>Prioritization</td>
<td>Sugiyanto and Rochimah [60]</td>
</tr>
</tbody>
</table>

Table 3.2: Summary of the integration of ANP in Technology Applications.
3.3 Applications of The ANP in Other Fields

The analytic network process has been used in several research areas such as education, industrial, personal, engineering, and so on. The following table shows a summary for some works in these various areas.

<table>
<thead>
<tr>
<th>Article Title</th>
<th>ANP Purpose</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improving intellectual capital model using analytic network process</td>
<td>Prioritization</td>
<td>Wudhikarn et al. [61]</td>
</tr>
<tr>
<td>Prioritizing Local Agenda 21 Programmes using Analytic Network Process: A Spanish Case Study</td>
<td>Prioritization</td>
<td>Peris et al. [63]</td>
</tr>
<tr>
<td>An Analytic Network Process (ANP) Approach to the Project Portfolio Management for Organizational Sustainability</td>
<td>Evaluation</td>
<td>Turan et al. [64]</td>
</tr>
<tr>
<td>Aircraft Selection Using Analytic Network Process: A Case for Turkish Airlines</td>
<td>Selection</td>
<td>Ozdemir et al. [65]</td>
</tr>
<tr>
<td>The Use of Analytic Network Process for Risk Assessment in Production of Renewable Energy from Agriculture Biomass in Latvia</td>
<td>Assessment</td>
<td>Rivas et al. [66]</td>
</tr>
<tr>
<td>Integrating analytic network process and data envelopment analysis for efficiency measurement of Turkish commercial banks</td>
<td>Evaluation</td>
<td>Ozdemir, Asi [67]</td>
</tr>
<tr>
<td>The analytic network process for the banking sector: An approach to evaluate the creditability of emerging industries</td>
<td>Evaluation</td>
<td>Chen et al. [68]</td>
</tr>
<tr>
<td>Applying Analytic Network Process in Logistics Service Provider Selection AASA Case Study of The Industry Investing in Southeast Asia</td>
<td>Selection</td>
<td>Chen and Wu [70]</td>
</tr>
<tr>
<td>Use of analytic network process in vendor selection decisions</td>
<td>Selection</td>
<td>Bayazit, Ozdens [71]</td>
</tr>
<tr>
<td>Prioritizing of effective factors on development of medicinal plants cultivation using the analytic network process</td>
<td>Prioritization</td>
<td>Rassam et al. [72]</td>
</tr>
<tr>
<td>Evaluation Model of TPL Provider of Agricultural Products Basing on Analytic Network Process</td>
<td>Evaluation</td>
<td>Huazhen et al. [73]</td>
</tr>
<tr>
<td>A Guarantee Model Based on ANP for Implementing “A Plan for Educating and Training Outstanding Engineers”</td>
<td>Evaluation</td>
<td>Xiao et al. [74]</td>
</tr>
<tr>
<td>A Corporate Social Responsibility Framework for Mining Sector Using Analytic Network Process</td>
<td>Assessment</td>
<td>Okan et al. [75]</td>
</tr>
</tbody>
</table>

Table 3.3: Summary of the ANP in various Applications.
Chapter 4

Case Studies Set-up

This chapter presents the educational case study used in this thesis. At the beginning, we describe the methodology used and the propositions for each ANP use in XP practices. Then, we introduce the educational case study that was conducted at the University of Regina. This chapter also includes the project data, such as duration, communication methods, and the customer’s duties.

4.1 Methodology

As mentioned in chapter 1, this research aims to investigate how the ANP can be used in XP practices. What are the benefits of using the ANP with XP practices? In this study, the ANP is applied in several areas, as proposed in chapter 5. Each XP practice has a certain goal and uses that can be achieved by using the ANP. Therefore, each practice has its own research question. In addition, the case study methodology is the research methodology in this study. This is an academic environment case study, in which two teams will develop an XP project. One of the two teams will use the ANP in the nine areas mentioned in chapter 5, while the other team will follow the traditional XP method. This section contains various pieces of significant information, such as details about the project that was given to develop is explained with
the requirements written by both customer and students. Furthermore, this chapter includes information such as developers’ background, communication methods, customer’s responsibilities, and the duration of the development.

The case study methodology consists of five main components, as described in [76], [77] and [78], which are:

1. Research questions.
2. Research propositions.
3. Units of analysis.
4. The logic linking of the data to the propositions.
5. Criteria for interpreting the findings.

Component 4 is presented in chapter 5; however, the rest of the case study components are presented in chapter 4.

4.2 Case Study Design

Defining the research questions and propositions is a significant step in designing a case study as explained in [79] and [80]. In this thesis, the questions and propositions are written in two formats. The first format is the specific format. Since the ANP is used for different purposes with XP practices, each practice has its specific research questions and propositions. The second format involves more general research questions and propositions, with a greater emphasis on “how” and “why” the ANP is beneficial to apply in extreme practices.

4.2.1 Research Questions in General Format

In this study, there are three main questions, which are:
1. How does the ANP help XP developers to fulfill their objectives in the mentioned practices?

2. How does the ANP influence the development team’s communication and productivity?

3. For each XP practice, what are the advantages that can be obtained from applying the ANP?

4.2.2 Research Propositions in General Format

According to [80] the study propositions can be defined based on the researcher’s understandings of XP and ANP. The study propositions are like hypotheses that are written by a researcher pre-conducting a case study. These propositions might be confirmed at the end of the case study when concluding the findings. The following are the main propositions in this thesis:

1. The ANP catches significant criteria and alternatives that affect each proposed XP practice. Also, the results of using the ANP display the order of alternatives based on their importance.

2. The ANP aids in various decision making problems, such as prioritization, selection, and ranking when it is required in XP practices.

3. The ANP includes creative debate and enhances team communication.

4. The ANP clarifies conflicting perspectives between the development teams within a certain practice.
4.2.3 Specific Research Questions and Propositions for Each XP Practice

The ANP is used in five XP practices; therefore, each use has a specific research questions and propositions.

Planning Game:

In planning game practice, the ANP is used in two areas. First, the ANP is used in order to rank XP prioritization methods, which can be used to prioritize user stories. Second, the ANP is used in order to select the best estimation method, which can be used to estimate each user story. The below research questions provide more details concerning the study’s analysis:

1. How can the ANP help the XP team to rank XP prioritization methods in order to prioritize user stories?
2. How can the ANP help the XP team to select the appropriate estimation method in order to estimate each user story?
3. How does the ANP influence the development team’s communication and productivity?

The followings are the research propositions:

1. The ANP catches significant criteria and alternatives that affect the selection of the prioritization method.
2. The ANP supports the prioritization activity in planning game practice.
3. The ANP catches significant criteria and alternatives that affect the selection of the estimation method.
4. The ANP includes a creative debate and enhances team communication.
5. The ANP clarifies conflicting perspectives between the development teams when selecting prioritization and estimation methods.

Simple Design:
The main aim of ANP in simple design practice is to analyze how ANP might be used in order to select the simple design tool from among several alternatives. The below research questions give more details about the study analysis:

1. How can the ANP help the XP team to select the appropriate simple design tool?

2. How does the ANP influence the development team’s communication and productivity in simple design practice?

The followings are the research propositions:

1. The ANP catches significant criteria and alternatives that affect the selection of the simple design tool.

2. The ANP supports the selection activity in simple design practice.

3. The ANP includes creative debate and enhances team communication.

4. The ANP clarifies conflicting perspectives between the development teams when selecting the best simple design tool.

Pair Programming:
In pair programming practice, the ANP is used in two areas. First, the ANP is used in order to match the best pairs. Second, the ANP is used to emphasize the best rules for matching pairs in pair programming practice. The research questions below provide further details concerning the study analysis:

1. How can the ANP help the XP team to select the best pairs with respect to certain criteria?
2. How can the ANP help the XP team to select the appropriate rules for matching pairs?

3. How does the ANP influence the development team’s communication and productivity in pair programming practice?

The followings are the research propositions:

1. The ANP catches significant criteria and alternatives that affect the selection of the best pairs and the selection of the rules for matching pairs.

2. The ANP supports selection activity in pair programming practice.

3. The ANP includes creative debate and enhances team communication.

4. The ANP clarifies conflicting perspectives between the development teams when performing pair programming.

**Test-Driven Development:**

The main aim is to analyze how ANP might be used in order to determine the type of testing. Furthermore, the ANP is used to rank the release indicators based on their importance. The research questions below provide more details about the study analysis:

1. How can the ANP help the XP team to select the best testing method with respect to certain criteria?

2. How can the ANP help the XP team to rank release indicators?

3. How does the ANP influence the development team’s communication and productivity in testing practice?

The followings are the research propositions:
1. The ANP catches significant criteria and alternatives that affect the selection of the best testing method.

2. The ANP supports ranking activity in testing practice.

3. The ANP includes creative debate and enhances team communication when performing testing.

**Refactoring:**

In refactoring practice, the ANP is used in two areas. First, the ANP is used in order to rank the refactoring techniques based on the internal attributes. Second, the ANP is used in order to rank the refactoring techniques based on the external attributes. The research questions below provide further details concerning the study analysis:

1. What is the significance of engaging the ANP when applying refactoring?

2. How can refactoring patterns be ranked using the ANP?

3. How does the ANP influence the development team’s communication and productivity in the refactoring practice?

4. How can the development team reduce time when refactoring using the ANP?

The followings are the research propositions:

1. The ANP catches significant criteria and alternatives that affect refactoring patterns.

2. The ANP supports ranking and selection activities in the refactoring practice.

3. The ANP includes creative debate and enhances team communication.

4. The ANP focuses on the most valuable refactoring methods in order to increase the quality of the code.
5. The ANP clarifies conflicting perspectives between the development teams when performing refactoring.

4.3 Criteria for Interpretation

After determining the study propositions, the criteria for interpretation of the findings should be determined as well [81]. When the final findings are analyzed, these findings are compared to the propositions to decide if they match with each other or not. Therefore, the criteria for interpretation are:

P1:

- Evidence shows that for each XP practice, ANP introduces the criteria and alternative clusters, as well as their level of relation.

- The ANP findings are displayed precisely with an order for both alternatives and criteria.

P2:

- Evidence shows that applying the ANP in XP practices is simple and understandable.

P3:

- Evidence shows that ANP helps to create a debatable environment between the development teams, which supports greater knowledge sharing.

P4:

- Evidence indicates that ANP helps to ensure that everyone’s voice in the team is heard and clarifies conflicting perspectives between the development teams in each practice.
4.4 Unit of Analysis

As explained in [81] the unit of analysis is derived from the study research questions. In this study, each practice has a different focus. Concerning the planning game, the main focus is to rank the prioritization methods and to select the best estimation technique. Concerning pair programming, the focus is to choose the best pairs and to decide the rules for matching developers. Concerning testing, the main goal is to select the type of testing and to prioritize the release indicators. Concerning simple design, the main objective is to select the best design tool. Concerning refactoring, the main focus is to rank the refactoring techniques in terms of external and internal quality attributes. Therefore, the units of analysis for this case study are the target uses of the ANP, such as selecting, ranking, prioritizing, and the process of evaluation. In addition, another unit of analysis is the participants’ perspectives concerning the benefits of the ANP in terms of each practice. Thus, the design of this case study involves multiple cases with multiple units of analysis.

4.5 Data Collection and Sources

At the beginning of each use of the ANP in XP, we investigated the ANP’s benefits and ability by introducing the related criteria and XP areas. Data were collected by searching through previous studies and a literature review. Also, data triangulation was performed in order to increase the study validity.

The major data source for this research was an XP project that was conducted during the winter of 2016 at the University of Regina. Therefore, the data sources in this thesis are:

- Questionnaires that were given to students during the XP project’s development.
Archival records, such as study plans, from the students.

- Comments from the customer.
- Open-ended interviews with the students.

4.6 Questionnaires

After each use of the ANP in XP practices, a questionnaire was given to the participants in order to obtain the students’ perspectives and experiences. Each questionnaire includes questions about the use of ANP as a selecting and prioritizing technique applied in XP, and about the participants’ perspectives concerning the benefits to the development team and their satisfaction level. The students submitted their answers to the researcher and their answers were saved in Microsoft Word files.

4.7 Semi-Structured Interviews

After acquiring the ANP results for all XP practices, we organized semi-structured interviews with the participants. One part of these interviews involved asking open-ended questions about the participants’ general viewpoints about the ANP. Another part asked about the advantages and disadvantages of ANP in each XP practice. The last part asked about the best applications of the ANP among all mentioned XP areas. During this phase, the collected data was kept in the form of handwritten notes, which were later organized in a folder to be analyzed in order to arrive at conclusions.
4.8 Field Notes

When developing the XP project, we used field notes that were taken by the researcher. Furthermore, these notes were taken during discussions among the participants and also included comments from the customer. These field notes helped the researcher to arrive at the final results of applying the ANP to the XP project.

4.9 Academic Case Study

The case study was conducted during a 12-week semester in the winter of 2016 at the University of Regina. The case study included 12 graduate students from the University of Regina and one additional participant, a client. These students had intermediate knowledge of XP processes and practices, and different programming levels. The participants’ backgrounds included experience with various programming languages, such as C++, Java, and PHP. The participants were organized into two teams. The first team used the ANP method in order to make their decisions in the mentioned areas, while the second team followed the traditional XP method. Both teams were asked to develop a project called “Professors’ Availability Managing System,” complete with a set of requirements. The project was developed in five iterations, allowing two weeks for each. At the end of the project, the two teams implemented all system requirements. Helpful materials that focused on planning game practices were given to the participants in order to ensure their understanding. These materials dealt with estimating user stories, writing user stories, and making programming commitments. The ANP team was given white papers, several presentations, and other important materials about the ANP in order to allow them to apply it in their development. Team 1 practiced on several pairwise comparisons and increased their understanding of the ANP structure. At the end, the researcher handed out a survey to the participants in order to collect additional data concerning
4.9.1 Team Formation

At the beginning of the study, participants’ information, such as programming level and XP experience, was collected in order to ensure that the two teams were roughly equal in terms of skill level. The first team applied the ANP to facilitate their decision making in the areas mentioned in chapter 5, whereas the second team followed the traditional XP process in their decision making, which is based on deep discussions and voting. In the case of both teams, every two participants were matched together in order to practice the pair programming.

4.9.2 Development Tools

Both teams developed the project using Visual Studio 2013, using C# as the programming language. Team 1 created a database using data explorer available with Visual Studio and SQL queries to access the database. Team 2 used My SQL database embedded in Microsoft Visual Studio 2013 and ASP.NET. Also, both teams used whiteboard, CRC cards, and UML diagrams during the design phase.

4.9.3 Communication Tools

The participants met together several times during each iteration. Furthermore, the participants and the customer shared their questions and knowledge via Dropbox shared folders. In addition, the participants communicated with each other using phone calls, SMS messages, and emails.
4.9.4 Project Assigned to the Participants

The participants were asked to develop a project called “Professors’ Availability Managing System,” complete with a set of requirements. The system focuses on the educational environment to provide a civilized way to arrange appointments with professors. The “Professors’ Availability Managing System” could provide many services for professors, researchers, and students in order to improve communication between them. Users of the system include professors, students, researchers, and administrators, and its services enable the following:

- As a student I want to:
  - Create an ordinary account using the students ID to verify the students.
  - Search for suitable professor.
  - Select desirable available appointment on the professor’s schedule.
  - Receive automatic notification confirmation email.
  - Remove or updating appointments.
  - Receive automatic notification from the system in case of cancelled or deferred appointment by the professor.
  - Receive a reminder two hours before each appointment.
  - View and track all appointments easily by viewing his confirmed appointments table on his/her main page.

- As a Researcher I want to:
  - Create an account which must have at least the following:
    * Personal Information
    * Education
    * Position
* Experience
* Interest area, and
* Current researches

– Search for professors they needed based on many criteria such as department, speciality, availability, benefit, etc.

– Book an appointment according the previous results.

– Receive automatic notification confirmation email.

– Remove or update appointments.

– Receive automatic notification from the system in case of cancelled or deferred appointment by professors.

– Check professors’ profiles in order to find an appropriate professor for the desired research. Just for reading no need to book an appointment.

– View and track all appointments easily by viewing his confirmed appointments table on his/her main page.

• As a Professor I want to:

  – Create an account which must have at least the following:

    * Personal Information
    * Education
    * Position
    * Experience
    * Interest area, and
    * Current researches

  – Add, remove, or change availability for appointments easily.

  – Cancel or defer appointments easily.
– The system updates the professor’s appointment table in real time in case of booking, cancelling, or deferring.

– Post or remove advertisements about professor’s researches. Positions available under my supervision.

– Check researchers’ profile in order to choose the appropriate researcher by searching researchers database in order to contact one in interest.

– Contact selected researchers about the available research positions.

• As an Administrators want to:

  – Create an account.
  
  – Update information on the system such as adding or removing professors and departments.
  
  – Add Role to Users.
  
  – Maintaining the entire system and handling errors.

4.9.5 User Stories Summary

Tables 4.1, 4.2, 4.3, 4.4, and 4.5 summarize the number of user stories, the number of tasks, tasks have been done, and tasks have been postponed for both teams in each iteration.

Iteration 1

Table 4.1: User stories and tasks for iteration 1

<table>
<thead>
<tr>
<th></th>
<th>User Stories</th>
<th>Tasks Defined</th>
<th>Tasks Done</th>
<th>Tasks Postponed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team 1</td>
<td>6</td>
<td>22</td>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>Team 2</td>
<td>7</td>
<td>17</td>
<td>12</td>
<td>5</td>
</tr>
</tbody>
</table>

Iteration 2
Table 4.2: User stories and tasks for iteration 2

<table>
<thead>
<tr>
<th></th>
<th>User Stories</th>
<th>Tasks Defined</th>
<th>Tasks Done</th>
<th>Tasks Postponed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team 1</td>
<td>5</td>
<td>22</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>Team 2</td>
<td>10</td>
<td>29</td>
<td>27</td>
<td>2</td>
</tr>
</tbody>
</table>

Iteration 3

Table 4.3: User stories and tasks for iteration 3

<table>
<thead>
<tr>
<th></th>
<th>User Stories</th>
<th>Tasks Defined</th>
<th>Tasks Done</th>
<th>Tasks Postponed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team 1</td>
<td>5</td>
<td>28</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>Team 2</td>
<td>4</td>
<td>22</td>
<td>15</td>
<td>7</td>
</tr>
</tbody>
</table>

Iteration 4

Table 4.4: User stories and tasks for iteration 4

<table>
<thead>
<tr>
<th></th>
<th>User Stories</th>
<th>Tasks Defined</th>
<th>Tasks Done</th>
<th>Tasks Postponed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team 1</td>
<td>4</td>
<td>11</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>Team 2</td>
<td>4</td>
<td>18</td>
<td>13</td>
<td>5</td>
</tr>
</tbody>
</table>

Iteration 5

Table 4.5: User stories and tasks for iteration 5

<table>
<thead>
<tr>
<th></th>
<th>User Stories</th>
<th>Tasks Defined</th>
<th>Tasks Done</th>
<th>Tasks Postponed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team 1</td>
<td>6</td>
<td>13</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>Team 2</td>
<td>4</td>
<td>14</td>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>

Technical Information

The tables 4.6 and 4.7 summarize the technical information of the two projects in each iteration.
**Team 1 Technical Information:**

*Table 4.6:* Team 1 technical information for each iteration.

<table>
<thead>
<tr>
<th></th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Iteration 4</th>
<th>Iteration 5</th>
</tr>
</thead>
<tbody>
<tr>
<td># Class in Packages</td>
<td>0</td>
<td>8</td>
<td>7</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td># Method in class</td>
<td>0</td>
<td>42</td>
<td>38</td>
<td>24</td>
<td>7</td>
</tr>
<tr>
<td>Line of Code in Class</td>
<td>1014</td>
<td>1980</td>
<td>1737</td>
<td>1407</td>
<td>850</td>
</tr>
</tbody>
</table>

**Team 2 Technical Information:**

*Table 4.7:* Team 2 technical information for each iteration.

<table>
<thead>
<tr>
<th></th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Iteration 4</th>
<th>Iteration 5</th>
</tr>
</thead>
<tbody>
<tr>
<td># Class in Packages</td>
<td>0</td>
<td>3</td>
<td>8</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td># Method in class</td>
<td>0</td>
<td>9</td>
<td>17</td>
<td>19</td>
<td>22</td>
</tr>
<tr>
<td>Line of Code in Class</td>
<td>732</td>
<td>1270</td>
<td>1658</td>
<td>2378</td>
<td>2734</td>
</tr>
</tbody>
</table>
Chapter 5

Applying ANP into Extreme Programming Practices

This chapter will address introducing the ANP into XP practices. The ANP is applied to several XP practices, but not all of them. The reason for this is because the other practices are more descriptive, which makes it unnecessary to investigate the decision-making process in these practices. In this thesis, we focus on five extreme practices in order to apply the ANP to the decision-making process. The selected practices are planning game, pair programming, simple design, refactoring, and test-driven development. In this chapter, each section introduces XP practice with its ANP application in order to investigate the benefits that ANP offers with regards to each practice.

5.1 ANP in Planning Game

In this section, two applications of ANP in planning game practice are presented. Furthermore, this section introduces activities that are related to planning game practice, such as user stories estimation and prioritization methods. In addition, it introduces the use of ANP in order to rank XP prioritization methods, and another
use of ANP to choose the best estimation method. A network structure is shown for each use, including a set of criteria and alternatives in order to facilitate the decision-making process. At the end of this section, the findings of the case study are described and discussed.

5.1.1 User Stories Overview

User stories provide short descriptions of tasks or functions from the point of view of system customers. According to O’heocha, “user stories are a common practice in agile methods for feeding user requirements into the development process. Unlike traditional requirements in engineering approaches, they do not call for comprehensive specification of the solution ‘up-front’ but instead encourage rich dialogue between customers and the technical team at implementation time to arrive at the best solution” [82]. Despite the fact that XP is an incremental approach, which makes it acceptable for changing user stories later in the design, these user stories should be considered carefully in order to structure the desired software. In addition, writing user stories can be affected by several issues related to the structure of user stories, including how much detail each story should have and their format, especially in the case of inexperienced customers.

According to Cohn, writing appropriate user stories should include the following [83]:

- User stories should be independent because, as Cohn states, “dependencies between stories lead to prioritization and planning problems... dependencies between stories can also make estimation much harder than it needs to be” [83].

- User stories should be debatable. In other words, “story cards are short descriptions of functionality, the details of which are to be negotiated in a conversation between the customer and the development team. Because story cards
are reminders to have a conversation rather than fully detailed requirements themselves, they do not need to include all relevant details” [83].

- User stories should be valuable. This means that they should have value for their users, and “the best way to ensure that each story is valuable to customers or users is to have the customer write the stories” [83].

- User stories should be estimable. For developers, it is important to be able to estimate each story, and to know how much time it might take to code each story. According to Cohn, developers might not be able to estimate a story because “the developers may lack domain knowledge. If the developers do not understand a story as it is written, they should discuss it with the customer who wrote the story” [83]. Another factor that may lead to an inability to estimate a story is the inexperience of developers. This means that they do not have adequate knowledge about technical issues and, therefore, cannot estimate a story [83].

- The size of user stories should be acceptable. This means that they should not be either too big or too small, in order to able to use the stories in planning [83]. Also, development teams have the authority to determine the appropriate size of stories either by splitting big stories into smaller ones or combining small ones into bigger ones [83].

- User stories should be able to be evaluated. In other words, they should be testable, in order to measure their appropriateness. Testable stories allow developers to know that the coding has been done.

The most significant factor in writing user stories, however, is their format. Formats differentiate user stories from traditional requirements; therefore, a story should be written in a specific format. According to O’heocha, “it should
be written in the form of a story. A format commonly used by agile teams takes the form” [82]:

‘As a <role> I want to <action> so that <result>’.

For example, we might write that as a student I want to access the university website so that I can register for courses.

5.1.2 Features Influencing Requirements Prioritization

Requirements can be prioritized based on various features. These features have no consensus about their importances in specifying the prioritization of the requirements. Also, these features seek to increase the delivered value to the user by making the most suitable decision.

Based on a survey by Wohlin and Aurum [84], Hoff et al. [81] introduced other features that influence decision making when specifying the requirement priorities. According to Wohlin and Aurum [84] factors such as delivery dates, the stakeholder priority of requirement, and development cost-benefit were found to be the most significant features during requirement prioritization. Hoff et al. [81] presented features such as impact of maintenance, complexity, increased performance, and cost-benefit to the organization. In addition, probability of success, testability, impact on the organization, and prior errors addressed are other factors added by Hoff et al. [81]. The authors investigated which features were the most significant by conducting a comprehensive survey. At the end of their study, the authors addressed the most significant features during the prioritizing of system requirements for implementation. These factors were complexity, cost-benefit to the organization, delivery data/schedule, requirement dependences, and fixes errors.

When prioritizing system requirements, the cost of requirement implementation was considered the most important feature by Bhoem et al. [85]. The cost of implementation involves various aspects, such as quality, documentation, stable require-
ments, availability of reusable software, complexity, and time frame.

Other factors that affect prioritizing requirements have been introduced by Fire-smith [86]. These factors include risk, time to market, personal preferences, requirements stability, legal mandate, dependencies, difficulty, business value, type of requirement, and frequency of use.

Bakalova et al. [87] proposed various factors that should be acknowledged when determining requirements prioritization. These factors include the effort required to perform measurement estimation regarding size, input from developers, the context of the project, associated dependencies, the external changes, and criteria regarding prioritization. The authors concentrated on business value, negative value, and risk, which are estimated by the user for the prioritization criteria.

Patel and Ramachandran [88] ranked user stories based on market value, business risk, business functionality, customer priority, core value, and implementation cost. Meanwhile, Wieger [89] prioritized the requirements’ importance according to risks associated with the implementation, the system benefits, technical cost, and penalties.

Carlshamre et al. [90] discussed requirement interdependencies by conducting an in-depth study. The authors presented the requirement interdependencies within various sets of requirements. The findings showed that 20% of the requirements are responsible for more than 70% of the interdependencies. Also, the authors noted that requirement interdependencies should be considered the most important factor when prioritizing requirements.

5.1.3 Applying ANP in Planning Game

As mentioned above, the ANP was used in two areas in planning game practice. The first use was to rank the XP prioritization techniques. The second use was choosing the appropriate estimation method.
5.1.4 First: Ranking the Prioritization Methods

In this section, a prioritization method is presented in order to help the development team to select the best method for ranking the user stories.

Introduction

The ANP is a well-structured tool that can help an XP team to rank prioritization techniques based on several criteria. Usually, stakeholders choose a well-known ranking method that is used by the development team, but this choice is usually not based on a formal approach. Well-known methods include the numeral assignment technique, weighted criteria analysis, binary search tree, requirements triage, dot voting, pair-wise analysis, top-ten requirements, and the Kano model.

In this section, the ANP is used to formalize the process of ranking five XP prioritization techniques, alternatives. These alternatives are the Kano model, relative weighting, top-ten requirements, 100-dollar test, and MoSCoW.

Prioritization Techniques Overview

There are several prioritization techniques that can be used to prioritize the user stories. In this thesis, the most commonly used methods are selected as alternatives, which might be summarized as follows:

1. Top-Ten Requirements:

   This method involves selecting ten requirements that are considered to be the most important by customers while ignoring the internal order of the selected requirements [91]. This ignorance is significant for the resolution of any conflict between the customers. Any one stakeholder can reach more than ten main requirements, but the challenge of this technique is that some stakeholders might not be able to specify their top priorities. This technique is more appropriate
2. **Cumulative Voting (The 100-Dollar Test)**

The 100-dollar test technique was explained by Leffingwell and Widrig [92] and can be used in order to prioritize requirements. This technique is simple and straightforward. The stakeholders receive 100 imaginary units (money, hours, etc.) to spread among the requirements. Regnell *et al.* [93] suggested using the amount of $100 units (1,000, 10,000 or 100,000) if there are too many requirements, in order to give the stakeholders more freedom in terms of the prioritizations. Stakeholders count the total for each requirement after spreading the units across the requirements, then prioritize the requirements based on the total units given to each.

3. **Relative Weighting**

This method assesses each requirement according to the impact of its being present or absent from the project. Each requirement is evaluated using a scale of 0 to 9, where 0 indicates low influence and 9 indicates a high influence. Each feature is given a value by the stakeholders for having it as well as a penalty for not having it. Then, the stakeholders count the value of each requirement as contributive to the entire requirements in order to obtain the relative value. Similarly, the stakeholders evaluate the cost for each requirement in relation to the entire requirements in order to obtain the relative cost. In the end, prioritization is decided upon by dividing the relative value by the relative cost [83].

4. **Kano Model**

In 1987, the Kano method was developed by Noriako Kano in order to organize the requirements into five groups based on answers to two questions [94]:
(a) “Functional question: How do you feel if this feature is present?”

(b) “Dysfunctional question: How do you feel if this feature in NOT present?”

From the five below options, the customer has to select one answer for each question:

(a) I like it.
(b) I expect it.
(c) I’m natural.
(d) I can tolerate it.
(e) I dislike it.

5. **MoSCoW**

This method can be used to prioritize the requirements based on their values from the customer’s point of view. The requirements are organized into four categories, as follows:

- **M**: Must have this attribute. This is not negotiable, and without it the project is considered a failure.
- **S**: Should have this attribute. Should be present if possible, in order to satisfy the customer. However, the project is not considered a failure in the case of its absence.
- **C**: Could have this attribute; however, it does not influence anything else. This is less critical, but it is nice to have it.
- **W**: Will not have it now, but would like to have it in the future.

**Proposed Criteria for Ranking the Prioritization Methods**

To rank the XP prioritization methods, it is necessary to identify the criteria that affect the ranking process. These criteria are compared with each other to show
the interdependencies and, also, are compared with respect to each alternative or prioritization technique. The prioritization techniques are compared with respect to the criteria in order to show the feedback relation in the ranking process. In this thesis, four criteria are proposed for ranking the prioritization techniques; however, different studies might apply the same methodology using different criteria. In this case study, four prioritization criteria were proposed. These criteria are:

1. Accuracy: Which prioritization technique provides the most accurate outcomes?

2. Simplicity: Which is the simplest prioritization method to understand and to apply?

3. Collaboration: Which prioritization method has the highest degree of collaboration between the team members?

4. Time: Which prioritization method saves the most time when prioritizing the user stories?

**ANP Structure for Prioritization Methods**

Structuring the problem in a network is the first step in the ANP. The network consists of three clusters. The objective cluster in the network ranks the prioritization methods. The second cluster is the criteria, which includes the following nodes: accuracy, simplicity, time and collaboration. The third cluster is alternatives, which includes the following nodes: top-ten requirements, MoSCoW, relative weighting, Kano model, and 100-dollar test.

Figure 5.1 shows the ANP network for ranking the prioritization techniques.

**Pairwise Comparisons for the Prioritization Methods**

The ANP team was asked to evaluate the importance of each criterion with respect to all of the prioritization techniques, in order to show the interdependencies. In
addition, the team was asked to evaluate each prioritization technique with respect to all of the criteria, in order to show the feedback of the network. The team 1 participants received the suitable ANP tables in order to make it easy for the participants to perform the pairwise comparisons based on the ANP fundamental scale that was described previously. For the interdependencies relationships, the participants were asked the following:

- With respect to top-ten requirements: which criterion is more important, simplicity or accuracy, and by how much?

- With respect to Top-Ten Requirements: which criterion is more important, accuracy or time and by how much?

- With respect to Top-Ten Requirements: which criterion is more important, collaboration or simplicity and by how much?

- With respect to Top-Ten Requirements: which criterion is more important, time or collaboration and by how much?

- With respect to Top-Ten Requirements: which criterion is more important, time
or simplicity and by how much?

• With respect to Top-Ten Requirements: which criterion is more important, accuracy or collaboration and by how much?

• With respect to MoSCoW: which criterion is more important, simplicity or accuracy and by how much?

• With respect to MoSCoW: which criterion is more important, accuracy or time and by how much?

• With respect to MoSCoW: which criterion is more important, collaboration or simplicity and by how much?

• With respect to MoSCoW: which criterion is more important, time or collaboration and by how much?

• With respect to MoSCoW: which criterion is more important, time or simplicity and by how much?

• With respect to MoSCoW: which criterion is more important, accuracy or collaboration and by how much?

• With respect to Relative Weighting: which criterion is more important, simplicity or accuracy and by how much?

• With respect to Relative Weighting: which criterion is more important, accuracy or time and by how much?

• With respect to Relative Weighting: which criterion is more important, collaboration or simplicity and by how much?

• With respect to Relative Weighting: which criterion is more important, time or collaboration and by how much?
• With respect to Relative Weighting: which criterion is more important, time or simplicity and by how much?

• With respect to Relative Weighting: which criterion is more important, accuracy or collaboration and by how much?

• With respect to Kano Model: which criterion is more important, simplicity or accuracy and by how much?

• With respect to Kano Model: which criterion is more important, accuracy or time and by how much?

• With respect to Kano Model: which criterion is more important, collaboration or simplicity and by how much?

• With respect to Kano Model: which criterion is more important, time or collaboration and by how much?

• With respect to Kano Model: which criterion is more important, time or simplicity and by how much?

• With respect to Kano Model: which criterion is more important, accuracy or collaboration and by how much?

• With respect to 100-Dollar Test: which criterion is more important, simplicity or accuracy and by how much?

• With respect to 100-Dollar Test: which criterion is more important, accuracy or time and by how much?

• With respect to 100-Dollar Test: which criterion is more important, collaboration or simplicity and by how much?
• With respect to 100-Dollar Test: which criterion is more important, time or collaboration and by how much?

• With respect to 100-Dollar Test: which criterion is more important, time or simplicity and by how much?

• With respect to 100-Dollar Test: which criterion is more important, accuracy or collaboration and by how much?

The participants then compared the prioritization methods with respect to each criterion in order to evaluate the feedback relationship. Examples of these questions for the participants are:

• Regarding simplicity: which method do you prefer, the Kano Model or MoSCoW?

• Regarding simplicity: which method do you prefer, the Kano Model or Top-Ten?

• Regarding simplicity: which method do you prefer, the Kano Model or 100-Dollar Test?

• Regarding simplicity: which method do you prefer, the Kano Model or Relative Weighting?

• Regarding simplicity: which method do you prefer, MoSCoW or Top-Ten?

• Regarding simplicity: which method do you prefer, MoSCoW or 100-Dollar Test?

• Regarding simplicity: which method do you prefer, MoSCoW or Relative Weighting?

• Regarding simplicity: which method do you prefer, Top-Ten or 100-Dollar Test?

• Regarding simplicity: which method do you prefer, Top-Ten or Relative Weighting?
• Regarding simplicity: which method do you prefer, 100-Dollar Test or Relative Weighting?

• Regarding accuracy: which method do you prefer, the Kano Model or MoSCoW?

• Regarding accuracy: which method do you prefer, the Kano Model or Top-Ten?

• Regarding accuracy: which method do you prefer, the Kano Model or 100-Dollar Test?

• Regarding accuracy: which method do you prefer, the Kano Model or Relative Weighting?

• Regarding accuracy: which method do you prefer, MoSCoW or Top-Ten?

• Regarding accuracy: which method do you prefer, MoSCoW or 100-Dollar Test?

• Regarding accuracy: which method do you prefer, MoSCoW or Relative Weighting?

• Regarding accuracy: which method do you prefer, Top-Ten or 100-Dollar Test?

• Regarding accuracy: which method do you prefer, Top-Ten or Relative Weighting?

• Regarding accuracy: which method do you prefer, 100-Dollar Test or Relative Weighting?

• Regarding time: which method do you prefer, the Kano Model or MoSCoW?

• Regarding time: which method do you prefer, the Kano Model or Top-Ten?

• Regarding time: which method do you prefer, the Kano Model or 100-Dollar Test?
• Regarding time: which method do you prefer, the Kano Model or Relative Weighting?

• Regarding time: which method do you prefer, MoSCoW or Top-Ten?

• Regarding time: which method do you prefer, MoSCoW or 100-Dollar Test?

• Regarding time: which method do you prefer, MoSCoW or Relative Weighting?

• Regarding time: which method do you prefer, Top-Ten or 100-Dollar Test?

• Regarding time: which method do you prefer, Top-Ten or Relative Weighting?

• Regarding time: which method do you prefer, 100-Dollar Test or Relative Weighting?

• Regarding collaboration: which method do you prefer, the Kano Model or MoSCoW?

• Regarding collaboration: which method do you prefer, the Kano Model or Top-Ten?

• Regarding collaboration: which method do you prefer, the Kano Model or 100-Dollar Test?

• Regarding collaboration: which method do you prefer, the Kano Model or Relative Weighting?

• Regarding collaboration: which method do you prefer, MoSCoW or Top-Ten?

• Regarding collaboration: which method do you prefer, MoSCoW or 100-Dollar Test?

• Regarding collaboration: which method do you prefer, MoSCoW or Relative Weighting?
• Regarding collaboration: which method do you prefer, Top-Ten or 100-Dollar Test?

• Regarding collaboration: which method do you prefer, Top-Ten or Relative Weighting?

• Regarding collaboration: which method do you prefer, 100-Dollar Test or Relative Weighting?

Prioritization Methods Results Based on ANP Evaluation

Each participant in Team 1 evaluated the prioritization methods according to the mentioned criteria. The Super Decision Software [97] was used to count the aggregation results for the ANP team.

For Team 1, according to all of the criteria, the ranking for the prioritization methods was shown as follows: first: Kano model, second: top-ten requirements, third: relative weighting, fourth: MoSCoW, and fifth: 100-dollar test. Table 5.1 shows these results. Furthermore, the importance of each criterion based on all of the prioritization techniques was as follows: first: simplicity, second: collaboration, third: time, and fourth: accuracy. Figure 5.2 exhibits these findings.

Table 5.1: Prioritization methods ranking by Team 1

<table>
<thead>
<tr>
<th>Methods</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kano Model</td>
<td>43.23 %</td>
</tr>
<tr>
<td>Top-Ten Requirements</td>
<td>22.20 %</td>
</tr>
<tr>
<td>Relative Weighting</td>
<td>14.60 %</td>
</tr>
<tr>
<td>MoSCoW</td>
<td>10.70 %</td>
</tr>
<tr>
<td>100-Dollar Test</td>
<td>9.25 %</td>
</tr>
</tbody>
</table>

For Team 2, the participants were asked to follow the traditional method in their decision making and, therefore, they were asked to document each step in their process of making decisions in terms of how and why the decision was made. Team 2
results show that the MoSCoW technique was given the highest rank among the other prioritization techniques. Table 5.2 displays Team 2’s ranking of prioritization methods. In terms of the most important factor for ranking the prioritization techniques, Team 2 gave collaboration the top score. Table 5.3 shows the ranking of the criteria by Team 2.

Table 5.2: Prioritization methods ranking by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MoSCoW</td>
</tr>
<tr>
<td>2</td>
<td>Top-Ten Requirements</td>
</tr>
<tr>
<td>3</td>
<td>Kano Model</td>
</tr>
<tr>
<td>4</td>
<td>100-Dollar Test</td>
</tr>
<tr>
<td>5</td>
<td>Relative Weighting</td>
</tr>
</tbody>
</table>

Table 5.3: The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Collaboration</td>
</tr>
<tr>
<td>2</td>
<td>Time</td>
</tr>
<tr>
<td>3</td>
<td>Accuracy</td>
</tr>
<tr>
<td>4</td>
<td>Simplicity</td>
</tr>
</tbody>
</table>
Observations

1. With respect to all of the criteria, Team 1 ranked the Kano model technique as the highest prioritization technique. Also, Team 1 ranked the top-ten requirements technique in the second position. In addition, the relative weighting technique was ranked in the third position by team 1, followed by MoSCoW and 100-dollar test in fourth and fifth positions, respectively.

2. Team 2 ranked the MoSCoW technique as the highest prioritization technique based on the traditional method of XP. Furthermore, in terms of the most important criteria, the Team 2 members ranked the collaboration factor as of the highest importance, while Team 2 ranked simplicity as a less important factor.

3. Team 1 considered simplicity to be the most important factor, and the collaboration factor was in the second position.

4. Time and accuracy criteria were ranked in the third and fourth positions, respectively, by Team 1.

5. Considering each criterion individually, we can notice that the 100-dollar test technique was given the top score in terms of accuracy by Team 1. The Kano model was ranked the highest with respect to the time, simplicity, and collaboration criteria. Table 5.4 shows the rankings of all techniques with respect to each criterion.

6. Considering each prioritization technique individually, Team 1 ranked the collaboration factor as the most important with respect to the MoSCoW, relative weighting and 100-dollar techniques. Team 1 ranked the simplicity factor the highest, according to the Kano model and top-ten requirements techniques.
Table 5.5 shows the weight of each criterion with respect to each prioritization technique.

7. Regarding the most preferable prioritization technique in terms of accuracy, Team 2 ranked the Kano model method the highest.
Table 5.4: Prioritization methods with respect to each criterion for Team 1

<table>
<thead>
<tr>
<th>Methods</th>
<th>Simplicity</th>
<th>Methods</th>
<th>Accuracy</th>
<th>Methods</th>
<th>Collaboration</th>
<th>Methods</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kano Model</td>
<td>40.58 %</td>
<td>100-Dollar</td>
<td>48.01 %</td>
<td>Kano Model</td>
<td>46.95 %</td>
<td>Kano Model</td>
<td>54.21 %</td>
</tr>
<tr>
<td>Top-Ten</td>
<td>26.99 %</td>
<td>MoSCoW</td>
<td>17.33 %</td>
<td>Relative Weighting</td>
<td>26.21 %</td>
<td>Top-Ten</td>
<td>22.06 %</td>
</tr>
<tr>
<td>Relative Weighting</td>
<td>12.09 %</td>
<td>Top-Ten</td>
<td>15.89 %</td>
<td>Top-Ten</td>
<td>12.91 %</td>
<td>MoSCoW</td>
<td>14.29 %</td>
</tr>
<tr>
<td>MoSCoW</td>
<td>11.38 %</td>
<td>Relative Weighting</td>
<td>14.99 %</td>
<td>100-Dollar</td>
<td>8.17 %</td>
<td>Relative Weighting</td>
<td>5.71 %</td>
</tr>
<tr>
<td>100-Dollar</td>
<td>8.94 %</td>
<td>Kano Model</td>
<td>3.75 %</td>
<td>MoSCoW</td>
<td>5.75 %</td>
<td>100-Dollar</td>
<td>3.70 %</td>
</tr>
</tbody>
</table>

Table 5.5: Criteria weights with respect to each alternative for Team 1

<table>
<thead>
<tr>
<th></th>
<th>Kano Model</th>
<th>Top-Ten</th>
<th>Relative Weighting</th>
<th>MoSCoW</th>
<th>100-Dollar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplicity</td>
<td>69.20 %</td>
<td>65.52 %</td>
<td>27.92 %</td>
<td>27.02 %</td>
<td>24.82 %</td>
</tr>
<tr>
<td>Collaboration</td>
<td>6.06 %</td>
<td>9.36 %</td>
<td>59.37 %</td>
<td>54.58 %</td>
<td>62.65 %</td>
</tr>
<tr>
<td>Time</td>
<td>21.90 %</td>
<td>20.93 %</td>
<td>9.34 %</td>
<td>12.20 %</td>
<td>8.84 %</td>
</tr>
<tr>
<td>Accuracy</td>
<td>2.83 %</td>
<td>4.18 %</td>
<td>3.35 %</td>
<td>6.18 %</td>
<td>3.66 %</td>
</tr>
</tbody>
</table>
5.1.5 Second: Ranking the Estimation Methods

In this section, a ranking method is presented in order to assist the development team in selecting the best method for estimating the user stories.

Introduction

During iteration planning meetings, estimates are used to establish the iteration plan. This occurs by assigning stories to each iteration based on their priorities. Using story points, the development team evaluates user stories to specify the cost and complexity of the implementation. After that, developers break down the user stories into small tasks.

There are several methods that can be used in software estimation. Some methods mathematically obtain an estimate by concentrating on historical data [98]. Other methods estimate effort by measuring task size [98]. Expert opinion works by asking an expert about each story. Based on his/her experience, the expert then gives an estimate. Another technique that can be used to estimate user stories is analogy. This technique can involve a triangulation process, where the story that is being estimated is compared with two other stories [99]. Disaggregation is another method that is used to split large stories into smaller ones, in order to estimate them.

Some studies suggest that the best technique that can be used to estimate stories is planning poker. This merges the three above-mentioned techniques of expert opinion, analogy, and disaggregation. All team members are involved in planning poker in order to estimate stories. This involves several steps and stages that are explained in more detail in [99].

In [98] the authors introduced several areas that are useful to estimate using expert opinion. These areas are: 1) areas where it is difficult to find empirical data, and 2) areas where it is difficult to estimate due to a lack of understanding regarding problems [98]. These mentions areas are considered reasons for widely applying the
expert opinion method in software estimation. In [98] the authors conducted an industrial experiment in order to evaluate the reliability of using the expert opinion method in cost estimation. The study aimed to enhance cost estimation in a medium-sized software organization. According to Faria and Miranda, “the intention of the organization in supporting this study was to assess its cost estimation capability in the bid phase of a project and, if required, to use the results as a catalyst for change of their estimation practices”[98]. In this study, an on-line survey was distributed among thirty employees. The response rate for the survey was 47%, and 7.5 years was the average experience of the participants. The research questions in this study focused on two areas: variability and calibration. The study contained three research questions asking about several issues; for example, the authors asked about the possibility of receiving the same estimate from various groups of estimators. After presenting the results, the authors concluded that the expert opinion method’s obstacles are inconsistencies and overconfidence. These issues cause unreliable software estimation. However, the authors suggested integrating the expert opinion method with another technique, such as Wideband-Delphi, in order to make expert opinion estimation more reliable.

Heemstra [100] conducted a survey of 364 organizations and found that only 51 estimated efforts used models. However, “the model users made no better estimate than the non-model users”[100]. The researcher concluded that expert opinion was better to use than estimation models.

In [101] the authors investigated effort estimation in 32 software projects from various Iranian software companies. A questionnaire was distributed in order to collect data from these companies concerning their project estimation. The study showed that most of these companies depend on small teams (five people or less) because of the small scale of developed products, and the simplicity involved in managing small teams. Furthermore, the most-used development processes in product development
are the rational unified process and XP. The distributed questionnaire included various questions, such as asking about the type of estimation method used in a project and the degree of difference between the estimated data and the real data [101]. Based on this study, expert opinion and the analogy method are the most common estimation methods. Among all these projects, 29% and 25% used the analogy method and expert opinion, respectively [101]. The study concluded that these two methods are simple to apply to similar projects and provide rapid estimation. However, the major defect is the dependence on expert judgment, which may be less reliable.

In [102] the authors conducted a study in order to evaluate the accuracy of the planning poker estimation method. A total of 13 students were formed into teams in order to develop “a web-based student records information system” [102]. All students received the same user stories and were asked to implement them in three sprints. Also, by using the planning poker method, students estimated the user stories and “the estimates provided by each team member during the first round were averaged to obtain the statistical combination for further comparison” [102]. At the same time, a number of experts were given the same user stories in order to provide their estimations. The results of this study showed that the planning poker method led the students to offer overly optimistic estimations, while the experts’ estimations were closer to the actual effort [102]. The study noted that the planning poker method is less beneficial when it is applied by less experienced developers.

Williams et al. [103] investigated XP practice development in an IBM group. They concluded that the XP product has improved the quality of pre-release and post release. The XP team noted enhancements in their effort estimation, schedule and productivity. Also, customer satisfaction was very high with the XP product, since developers delivered more than was planned.

Finnie and Wittig [104] estimated effort by applying artificial neural network (ANN) and case-based reasoning. They concluded that by “using a data set from
the Australian Software Metrics Association, ANN was able to estimate development effort within 25% of the actual effort in more than 75% of the projects”[104].

In [105] the authors emphasized several challenges concerning current estimation methods. For example, “it cannot be easily related to the time duration because story points represent the amount of work, and the velocity differs from team to team”[105].

Proposed Criteria for Ranking the Estimation Method

To select the best XP estimation method, it is necessary to identify the criteria that influence the estimation process. These criteria are compared with each other to show the interdependencies and, also, they are compared with respect to each alternative or estimation method. The estimation methods are compared with respect to the criteria in order to show the feedback relation in the selection process. In this case study, four estimation criteria were proposed. These criteria are:

1. Accuracy: Which estimation technique gives the most accurate estimation?
2. Simplicity: What is the simplest estimation method to understand and to apply?
3. Collaboration: Which estimation method has the highest degree of collaboration between the team members?
4. Time: Which estimation method saves the most time when estimating the user stories?

ANP Structure for Estimation Methods

Structuring the problem as a network is the first step in the ANP. The ANP network contains the criteria cluster, alternative cluster, and the goal. The main cluster is the alternative cluster, which contains the four estimation methods where the desire is to select the best from among them. The three network components are: the goal
cluster-selecting the best estimation method; the criteria cluster-accuracy, simplicity, collaboration, and time; the alternatives cluster-analogy, expert opinion, disaggregation, and planning poker. Figure 5.3 illustrates the ANP structure used in this paper.

![ANP structure for selecting the best estimation method](image)

**Figure 5.3:** ANP structure for selecting the best estimation method

**Pairwise Comparisons for the Estimation Methods**

The ANP team was asked to evaluate the importance of each criterion according to all of the estimation techniques in order to show the interdependencies. In addition, the team was asked to evaluate each prioritization technique with respect to all of the criteria in order to show the feedback of the network. The Team 1 participants received the suitable ANP tables, in order to make it easy for the participants to perform the pairwise comparisons based on the ANP fundamental scale that was described previously. For the interdependencies relationships, the participants were asked the following:

- With respect to Planning Poker: which criterion is more important, simplicity or accuracy and by how much?
• With respect to Planning Poker: which criterion is more important, accuracy or time and by how much?

• With respect to Planning Poker: which criterion is more important, collaboration or simplicity and by how much?

• With respect to Planning Poker: which criterion is more important, time or collaboration and by how much?

• With respect to Planning Poker: which criterion is more important, time or simplicity and by how much?

• With respect to Planning Poker: which criterion is more important, accuracy or collaboration and by how much?

• With respect to Analogy: which criterion is more important, simplicity or accuracy and by how much?

• With respect to Analogy: which criterion is more important, accuracy or time and by how much?

• With respect to Analogy: which criterion is more important, collaboration or simplicity and by how much?

• With respect to Analogy: which criterion is more important, time or collaboration and by how much?

• With respect to Analogy: which criterion is more important, time or simplicity and by how much?

• With respect to Analogy: which criterion is more important, accuracy or collaboration and by how much?
• With respect to Expert Opinion: which criterion is more important, simplicity or accuracy and by how much?

• With respect to Expert Opinion: which criterion is more important, accuracy or time and by how much?

• With respect to Expert Opinion: which criterion is more important, collaboration or simplicity and by how much?

• With respect to Expert Opinion: which criterion is more important, time or collaboration and by how much?

• With respect to Expert Opinion: which criterion is more important, time or simplicity and by how much?

• With respect to Expert Opinion: which criterion is more important, accuracy or collaboration and by how much?

• With respect to Disaggregation: which criterion is more important, simplicity or accuracy and by how much?

• With respect to Disaggregation: which criterion is more important, accuracy or time and by how much?

• With respect to Disaggregation: which criterion is more important, collaboration or simplicity and by how much?

• With respect to Disaggregation: which criterion is more important, time or collaboration and by how much?

• With respect to Disaggregation: which criterion is more important, time or simplicity and by how much?
With respect to Disaggregation: which criterion is more important, accuracy or collaboration and by how much?

The participants then compared the estimation methods with respect to each criterion, in order to evaluate the feedback relationship. Examples of these questions for the participants are:

- With respect to simplicity: which method do you prefer, Planning Poker or Analogy?
- With respect to simplicity: which method do you prefer, Planning Poker or Disaggregation?
- With respect to simplicity: which method do you prefer, Planning Poker or Expert Opinion?
- With respect to simplicity: which method do you prefer, Analogy or Disaggregation?
- With respect to simplicity: which method do you prefer, Analogy or Expert Opinion?
- With respect to simplicity: which method do you prefer, Disaggregation or Expert Opinion?
- With respect to accuracy: which method do you prefer, Planning Poker or Analogy?
- With respect to accuracy: which method do you prefer, Planning Poker or Disaggregation?
- With respect to accuracy: which method do you prefer, Planning Poker or Expert Opinion?
• With respect to accuracy: which method do you prefer, Analogy or Disaggregation?

• With respect to accuracy: which method do you prefer, Analogy or Expert Opinion?

• With respect to accuracy: which method do you prefer, Disaggregation or Expert Opinion?

• With respect to time: which method do you prefer, Planning Poker or Analogy?

• With respect to time: which method do you prefer, Planning Poker or Disaggregation?

• With respect to time: which method do you prefer, Planning Poker or Expert Opinion?

• With respect to time: which method do you prefer, Analogy or Disaggregation?

• With respect to time: which method do you prefer, Analogy or Expert Opinion?

• With respect to time: which method do you prefer, Disaggregation or Expert Opinion?

• With respect to collaboration: which method do you prefer, Planning Poker or Analogy?

• With respect to collaboration: which method do you prefer, Planning Poker or Disaggregation?

• With respect to collaboration: which method do you prefer, Planning Poker or Expert Opinion?

• With respect to collaboration: which method do you prefer, Analogy or Disaggregation?
With respect to collaboration: which method do you prefer, Analogy or Expert Opinion?

With respect to collaboration: which method do you prefer, Disaggregation or Expert Opinion?

Estimation Methods Results Based on ANP Evaluation

In Team 1, each participant individually evaluated the estimation techniques using the ANP pairwise comparisons. Super Decisions Software was used in order to determine the aggregation judgments for team 1.

Team 1's results show that the expert opinion method was given the highest rank among the four alternatives. Planning poker came second, followed by analogy, then disaggregation. Table 5.6 exhibits the relative weight of each one as a percentage.

In addition, by using the software, we are able to examine the significance of each criterion based on all of the estimation techniques. The accuracy criterion was ranked as being of the highest importance among the criteria, followed by collaboration, simplicity, and time. Figure 5.4 shows the criteria importance scores.

**Table 5.6:** Estimation methods ranking for Team 1

<table>
<thead>
<tr>
<th>Methods</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert Opinion</td>
<td>35.36 %</td>
</tr>
<tr>
<td>Planning Poker</td>
<td>33.64 %</td>
</tr>
<tr>
<td>Analogy</td>
<td>25.50 %</td>
</tr>
<tr>
<td>Disaggregation</td>
<td>5.48 %</td>
</tr>
</tbody>
</table>

For Team 2, the participants were asked to follow the traditional method in their decision making and, therefore, were asked to document each step in their process in terms of how and why the decision was made. Most of their decisions were made based on deep discussions and voting. Team 2’s results show that the analogy method was given the highest rank among the estimation methods. Table 5.7 displays the
estimation methods ranking by Team 2. In terms of the most important factor for ranking the estimation methods, Team 2 gave accuracy the top score. Table 5.8 shows the ranking of the criteria by Team 2.

Table 5.7: Estimation methods ranking by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Analogy</td>
</tr>
<tr>
<td>2</td>
<td>Expert Opinion</td>
</tr>
<tr>
<td>3</td>
<td>Planning Poker</td>
</tr>
<tr>
<td>4</td>
<td>Disaggregation</td>
</tr>
</tbody>
</table>

Table 5.8: The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Accuracy</td>
</tr>
<tr>
<td>2</td>
<td>Time</td>
</tr>
<tr>
<td>3</td>
<td>Collaboration</td>
</tr>
<tr>
<td>4</td>
<td>Simplicity</td>
</tr>
</tbody>
</table>

Observations

1. With respect to all of the criteria, Team 1 ranked the expert opinion technique as the highest estimation technique. Furthermore, Team 1 ranked the planning
poker technique in the second position. In addition, the analogy technique was ranked in the third position by Team 1, followed by disaggregation in the last position.

2. Team 2 ranked the analogy technique as the highest estimation technique based on the traditional method of XP. Furthermore, in terms of the most important criteria, the Team 2 members ranked accuracy as of the highest importance, while Team 2 ranked simplicity as a less important factor.

3. Team 1 considered accuracy the most important factor, and collaboration was in the second position.

4. Simplicity and time criteria were ranked in the third and fourth positions, respectively, by Team 1.

5. Considering each criterion individually, we can notice that the planning poker technique was given the top score in terms of accuracy by Team 1. However, analogy was ranked the highest with respect to the collaboration criterion.

6. Expert opinion was ranked the highest by Team 1 in terms of the time and simplicity criteria. Table 5.9 shows the ranking for all techniques with respect to each criterion.

7. Considering each estimation technique individually, Team 1 ranked the accuracy factor as the most important with respect to the planning poker and expert opinion techniques. The time factor was ranked the highest by Team 1 according to the disaggregation technique. Table 5.10 shows the weight of each criterion with respect to each estimation technique.

8. Regarding the most preferable estimation technique in terms of time, Team 2 ranked expert opinion as the highest.
Table 5.9: Estimation methods with respect to each criterion for Team 1

<table>
<thead>
<tr>
<th>Methods</th>
<th>Simplicity</th>
<th>Methods</th>
<th>Accuracy</th>
<th>Methods</th>
<th>Collaboration</th>
<th>Methods</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert Opinion</td>
<td>54.23 %</td>
<td>Planning Poker</td>
<td>63.48 %</td>
<td>Analogy</td>
<td>51.04 %</td>
<td>Expert Opinion</td>
<td>63.43 %</td>
</tr>
<tr>
<td>Analogy</td>
<td>28.62 %</td>
<td>Expert Opinion</td>
<td>23.39 %</td>
<td>Expert Opinion</td>
<td>31.74 %</td>
<td>Analogy</td>
<td>20.17 %</td>
</tr>
<tr>
<td>Planning Poker</td>
<td>13.88 %</td>
<td>Disaggregation</td>
<td>7.62 %</td>
<td>Planning Poker</td>
<td>12.77 %</td>
<td>Planning Poker</td>
<td>12.61 %</td>
</tr>
<tr>
<td>Disaggregation</td>
<td>3.25 %</td>
<td>Analogy</td>
<td>5.50 %</td>
<td>Disaggregation</td>
<td>4.43 %</td>
<td>Disaggregation</td>
<td>3.78 %</td>
</tr>
</tbody>
</table>

Table 5.10: Criteria weights with respect to each alternative for Team 1

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Planning Poker</th>
<th>Expert Opinion</th>
<th>Analogy</th>
<th>Disaggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplicity</td>
<td>22.61 %</td>
<td>4.20 %</td>
<td>23.63 %</td>
<td>3.16 %</td>
</tr>
<tr>
<td>Collaboration</td>
<td>9.37 %</td>
<td>35.17 %</td>
<td>61.56 %</td>
<td>21.88 %</td>
</tr>
<tr>
<td>Time</td>
<td>4.57 %</td>
<td>10.27 %</td>
<td>10.24 %</td>
<td>64.92 %</td>
</tr>
<tr>
<td>Accuracy</td>
<td>63.44 %</td>
<td>50.34 %</td>
<td>4.56 %</td>
<td>10.02 %</td>
</tr>
</tbody>
</table>
5.2 ANP in Simple Design

In this section, the use of ANP in XP simple design practice is described. Also, this section presents the concept of simple design and its methods. This section also shows the ANP structure in simple design practice that includes the related criteria and the design tools, in order to select the best simple design tool. The results of the study are discussed and presented.

5.2.1 Introduction

The simple design practice reflects performing the job with the simplest design, and how developers should only design for today. According to Beck [19] the simple design is the one that contains no duplicated logic, passes all the tests, has least possible number of classes and methods, and establishes communication among team members. Others researchers have proposed different rules, such as the need to obtain high cohesion and contain loose coupling [106]. Simple design requires that any complexity in the code be removed immediately.

5.2.2 Selecting the Simple Design Tool

The ANP is used in simple design practice in order to select the best simple design tool to be used by the XP team. The selected design tools and their significant issues are summarized as follows:

Simple Design Tools:

Simple design helps to avoid the design problems associated with complex designs. For heavy-weight projects, however, more complex designs are considered essential [107]. Different project architectures give the XP team a variety of design practices to use. The XP team members have several design tools to select from, as follows:

1. Class Responsibility Collaboration (CRC cards) is a tool that can be used in the
design of software. Cunningham and Beck \[108\] have introduced the CRC cards as a teaching tool; however, these CRC cards were recommended by expert designers and XP developers. The CRC is described as a flexible and quick tool to determine the class with a set of its objects, its members, and the relationships within the design of the system. Each CRC card is divided into three parts, as follows \[109\]:

- A class represents a set of similar objects.
- A responsibility is what a class does or knows.
- A collaborator is another class that the class interacts with in order to complete the job.

2. Unified Modeling Language (UML): Software developers and engineers are able to create diagrams and architectures using UML as a powerful modeling tool. Often, the UML tool requires an initial understanding of a big design, and this may be at odds with the XP simple design.

However, researchers like Ambler \[110\] have studied the relationship between UML and XP. Ambler investigated this relation by asking two questions \[110\]:

(a) Can we use UML with XP?

(b) How do we use UML with XP?

Ambler \[110\] answered the first question affirmatively. An XP team can apply the UML artifacts during the XP development. However, he answered the second question by suggesting that this question was wrong and that it should be, “How do we model effectively on an XP project?” In the end, Ambler suggested minimizing the UML using XP.

3. Whiteboard or E-Whiteboard: This tool is preferred by some XP members, as they consider it an extremely simple method. Chen et al. \[111\] highlighted
several advantages of using whiteboard in software design; for example, immediacy, which means this tool is well known and it does not require much effort to make it available to developers. Also, using the whiteboard tool makes it easier to design diagrams and add or delete information. Another advantage is versatility. Using a whiteboard allows designers to catch multiple diagrams for different design notations and sub notations, such as arrows and comments. In addition, the whiteboard tool gives designers more flexibility by doing their job without being attention for formal manner. Chen et al. [111] explained other advantages of using whiteboard in software design as well, such as the size of this tool is often sufficiently large to capture various design sketches; additional whiteboards support collaboration among designers by allowing them to discuss evolving designs [110].

4. A combination of UML and CRC cards: Some studies, like Borstler [112], suggested the use of the combination of some UML elements and developed role-play diagrams (RPDs) and collaboration diagrams in order to be integrated with the CRC cards tool. Borstler [112] discovered that this combination encourages “objective thinking” and facilitates understanding in object-oriented programs. Nevertheless, the author acknowledges that the RPDs are nearer to Booch’s diagram than UML diagrams. According to Ambler [110] the UML is inadequate for the business applications that are developed following the XP method.

5.2.3 Proposed Criteria for Choosing the Design Method

In order to achieve the goal of selecting the design tool, the participants applied the ANP assessment steps explained previously. The participants evaluated the design method according to the following criteria:
• **Documentation**: Which design method produces the most valuable documentation?

• **Communication**: Which design method best assists communication among the team members?

• **Portability**: Which design method is the easiest to transport?

• **Simplicity**: What is the simplest design method to understand and to apply?

### 5.2.4 ANP Structure for Design Tools

The ANP structure for selecting the best design tool consists of three clusters. The main cluster is the goal of selecting the best design tool. The other two clusters exhibit the process of selecting the design tool based on the criteria, which are documentation, communication, portability, and simplicity, and the alternatives, which are UML, CRC cards, whiteboard, and the combined method. Figure 5.5 illustrates the ANP structure for the problem.

![ANP network for selecting the design method](image)

*Figure 5.5: ANP network for selecting the design method*
5.2.5 Pairwise Comparisons for the Design Methods

The ANP team was asked to evaluate the importance of each criterion with respect to all of the design tools, in order to exhibit the interdependencies. Additionally, the team was asked to evaluate each design tool with respect to all of the criteria, in order to show the feedback of the network. Team 1 participants received the suitable ANP tables in order to make it easy for the participants to do the pairwise comparisons based on the ANP fundamental scale that was described previously. For the interdependencies relationship, the participants were asked the following:

- With respect to UML: which criterion is more important, simplicity or communication and by how much?
- With respect to UML: which criterion is more important, communication or documentation and by how much?
- With respect to UML: which criterion is more important, portability or simplicity and by how much?
- With respect to UML: which criterion is more important, documentation or portability and by how much?
- With respect to UML: which criterion is more important, documentation or simplicity and by how much?
- With respect to UML: which criterion is more important, communication or portability and by how much?
- With respect to CRC Cards: which criterion is more important, simplicity or communication and by how much?
- With respect to CRC Cards: which criterion is more important, communication or documentation and by how much?
• With respect to CRC Cards: which criterion is more important, portability or simplicity and by how much?

• With respect to CRC Cards: which criterion is more important, documentation or portability and by how much?

• With respect to CRC Cards: which criterion is more important, documentation or simplicity and by how much?

• With respect to CRC Cards: which criterion is more important, communication or portability and by how much?

• With respect to Combination Methods: which criterion is more important, simplicity or communication and by how much?

• With respect to Combination Methods: which criterion is more important, communication or documentation and by how much?

• With respect to Combination Methods: which criterion is more important, communication or portability and by how much?

• With respect to Combination Methods: which criterion is more important, portability or simplicity and by how much?

• With respect to Combination Methods: which criterion is more important, documentation or portability and by how much?

• With respect to Combination Methods: which criterion is more important, documentation or simplicity and by how much?

• With respect to Combination Methods: which criterion is more important, communication or portability and by how much?

• With respect to Whiteboard: which criterion is more important, simplicity or communication and by how much?
• With respect to Whiteboard: which criterion is more important, communication or documentation and by how much?

• With respect to Whiteboard: which criterion is more important, portability or simplicity and by how much?

• With respect to Whiteboard: which criterion is more important, documentation or portability and by how much?

• With respect to Whiteboard: which criterion is more important, documentation or simplicity and by how much?

• With respect to Whiteboard: which criterion is more important, communication or portability and by how much?

The participants then compared the design tools with respect to each criterion, in order to evaluate the feedback relationship. Examples of these questions for the participants are:

• With respect to simplicity: which method do you prefer, UML or CRC Cards?

• With respect to simplicity: which method do you prefer, UML or Combination Methods?

• With respect to simplicity: which method do you prefer, UML or Whiteboard?

• With respect to simplicity: which method do you prefer, CRC Cards or Combination Methods?

• With respect to simplicity: which method do you prefer, CRC Cards or Whiteboard?

• With respect to simplicity: which method do you prefer, Combination Methods or Whiteboard?
• With respect to communication: which method do you prefer, UML or CRC Cards?

• With respect to communication: which method do you prefer, UML or Combination Methods?

• With respect to communication: which method do you prefer, UML or Whiteboard?

• With respect to communication: which method do you prefer, CRC Cards or Combination Methods?

• With respect to communication: which method do you prefer, CRC Cards or Whiteboard?

• With respect to communication: which method do you prefer, Combination Methods or Whiteboard?

• With respect to documentation: which method do you prefer, UML or CRC Cards?

• With respect to documentation: which method do you prefer, UML or Combination Methods?

• With respect to documentation: which method do you prefer, UML or Whiteboard?

• With respect to documentation: which method do you prefer, CRC Cards or Combination Methods?

• With respect to documentation: which method do you prefer, CRC Cards or Whiteboard?
With respect to documentation: which method do you prefer, Combination Methods or Whiteboard?

With respect to portability: which method do you prefer, UML or CRC Cards?

With respect to portability: which method do you prefer, UML or Combination Methods?

With respect to portability: which method do you prefer, UML or Whiteboard?

With respect to portability: which method do you prefer, CRC Cards or Combination Methods?

With respect to portability: which method do you prefer, CRC Cards or Whiteboard?

With respect to portability: which method do you prefer, Combination Methods or Whiteboard?

5.2.6 Design Tools Results Based on ANP Evaluation

The ANP team ranked the whiteboard design tool at the top, followed by the UML tool. Team 1 ranked the combination method and CRC cards in the third and fourth positions, respectively. Table 5.11 summarizes the ANP team results. Also, figure 5.6 shows the importance of each criterion as a percentage, according to Team 1.

Table 5.11: Design methods ranking for Team 1

<table>
<thead>
<tr>
<th>Methods</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whiteboard</td>
<td>41.64 %</td>
</tr>
<tr>
<td>UML</td>
<td>23.50 %</td>
</tr>
<tr>
<td>Combined Methods</td>
<td>20.01 %</td>
</tr>
<tr>
<td>CRC Cards</td>
<td>14.83 %</td>
</tr>
</tbody>
</table>
Team 2’s results show that the UML method was given the highest rank among the design methods. Table 5.12 displays the simple design methods ranking by Team 2. In terms of the most important factor for ranking the design methods, Team 2 gave documentation the top score. Table 5.13 shows the ranking of the criteria by Team 2.

**Table 5.12: Design methods ranking by Team 2**

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UML</td>
</tr>
<tr>
<td>2</td>
<td>Whiteboard</td>
</tr>
<tr>
<td>3</td>
<td>Combined Methods</td>
</tr>
<tr>
<td>4</td>
<td>CRC Cards</td>
</tr>
</tbody>
</table>

**Table 5.13: The importance of the criteria by Team 2**

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Documentation</td>
</tr>
<tr>
<td>2</td>
<td>Portability</td>
</tr>
<tr>
<td>3</td>
<td>Communication</td>
</tr>
<tr>
<td>4</td>
<td>Simplicity</td>
</tr>
</tbody>
</table>

![Figure 5.6: The importance of the criteria by Team 1](image)
5.2.7 Observations

1. With respect to all criteria, Team 1 ranked the whiteboard tool as the highest design tool.

2. The UML tool was ranked in the second position by Team 1 followed by combined methods and CRC cards in third and fourth positions, respectively.

3. The UML tool was ranked as the highest design tool by Team 2.

4. The CRC cards were ranked in the lowest position by both teams.

5. Team 1 considered communication as the most important criterion.

6. In terms of criteria importance, Team 2 ranked documentation as the most important criterion.

7. Considering each criterion individually, we can notice that the whiteboard tool was given the top score in terms of communication, portability, and simplicity by Team 1. However, UML was ranked the highest with respect to the documentation criterion. Table 5.14 shows the ranking for all design tools with respect to each criterion.

8. Regarding the most preferable design tool in term of simplicity, Team 2 gave the whiteboard tool the top score.

9. Considering each design tool individually, Team 1 ranked documentation as the most important criterion with respect to UML and CRC cards tools. Team 1 ranked the communication criterion the highest with respect to the whiteboard tool. Table 5.15 shows the weight of each criterion with respect to each design technique.
Table 5.14: Design methods with respect to each criterion for Team 1

<table>
<thead>
<tr>
<th>Methods</th>
<th>Simplicity</th>
<th>Methods</th>
<th>Documentation</th>
<th>Methods</th>
<th>Communication</th>
<th>Methods</th>
<th>Portability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whiteboard</td>
<td>55.80 %</td>
<td>UML</td>
<td>62.97 %</td>
<td>Whiteboard</td>
<td>54.39 %</td>
<td>Whiteboard</td>
<td>58.36 %</td>
</tr>
<tr>
<td>CRC Cards</td>
<td>27.88 %</td>
<td>Combined Methods</td>
<td>23.63 %</td>
<td>Combined Methods</td>
<td>27.05 %</td>
<td>CRC Cards</td>
<td>23.43 %</td>
</tr>
<tr>
<td>Combined Methods</td>
<td>11.30 %</td>
<td>CRC Cards</td>
<td>8.46 %</td>
<td>UML</td>
<td>12.19 %</td>
<td>Combined Methods</td>
<td>13.03 %</td>
</tr>
<tr>
<td>UML</td>
<td>0.50 %</td>
<td>Whiteboard</td>
<td>4.91 %</td>
<td>CRC Cards</td>
<td>6.36 %</td>
<td>UML</td>
<td>5.25 %</td>
</tr>
</tbody>
</table>

Table 5.15: Criteria weights with respect to each design tool for Team 1

<table>
<thead>
<tr>
<th></th>
<th>Whiteboard</th>
<th>UML</th>
<th>CRC Cards</th>
<th>Combined Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplicity</td>
<td>4.43 %</td>
<td>28.07 %</td>
<td>3.85 %</td>
<td>61.25 %</td>
</tr>
<tr>
<td>Communication</td>
<td>53.86 %</td>
<td>13.47 %</td>
<td>9.45 %</td>
<td>21.56 %</td>
</tr>
<tr>
<td>Documentation</td>
<td>9.40 %</td>
<td>55.54 %</td>
<td>58.35 %</td>
<td>11.55 %</td>
</tr>
<tr>
<td>Portability</td>
<td>32.29 %</td>
<td>2.89 %</td>
<td>28.33 %</td>
<td>5.62 %</td>
</tr>
</tbody>
</table>
5.3 ANP in Pair Programming

In this section, two applications of ANP in pair programming practice are presented. This section also presents a brief description of pair programming and research related to this practice. In addition, it introduces the use of ANP in order to select the best pairs, and another use of ANP to rank the rules for matching pairs. A network structure is shown for each use, including a set of criteria and alternatives, in order to facilitate the decision making process. The end of this section shows and discusses the findings of the case study and its results.

5.3.1 Introduction

Pair programming practice involves two programmers working on a single machine to perform one task. The programmer who types the code is called the driver, and the second programmer, who is responsible for monitoring the current work, is called the navigator [113]. Several benefits can be achieved through pair programming, such as increased team productivity, fewer defects, enhanced system quality, high customer satisfaction, provision of rapid feedback, reduced code and better design, shared knowledge, and better team communication [114].

VanDeGrift [115] discovered that pair programming helps increase confidence and improves the programmer’s performance. The author emphasized that pair programming assists partners in understanding the project solution. Additionally, it sometimes minimizes the level of frustration. Another benefit is that pair programming can be an opportunity to enhance programmers’ skills and teach programming.

Katira et al. [116] investigated compatibility among programmers with different skills in order to develop guidelines for matching student pairs. This investigation included freshman, advanced undergraduate, and graduate students. The findings showed that students’ attention to their partner’s level of skill has an important
effect on their compatibility. For example, both freshmen and graduate students work better with students at the same or a similar skill level.

At the University of Utah, Williams et al. [117] organized a study that involved advanced undergraduate students. The main objective of the study was to use qualitative and quantitative evidence in order to prove that pair programming generates high quality software in less time while increasing programmer confidence. Tomayko [118] validated that pair programming allows programmers to perform coding with fewer errors than individual programmers.

Chaparro et al. [119] conducted a study with a sample of postgraduate students that involved applying pair programming as an educational technique. The main objective of the study was to explore the features that may influence the success of pair programming. In their case study, the authors answered the question: Why is pair programming sometimes ineffective? By observing, recording and interviewing students who practiced pair programming, the authors found that the major features that influenced the success of pair programming were level of skill, assigned tasks, and changing roles.

Dick and Zamett [120] highlighted that different personal characteristics should be acknowledged when matching two programmers, in order to guarantee sufficient collaboration, convenience working with the partner, sufficient communication, and the ability to negotiate.

Katira et al. [121] carried out a study with a sample of 361 software engineering students at North Carolina State University to predict and understand pair compatibility. The authors found that participants were more compatible with partners whom they perceived as having a similar skill level. Midterm grades and GPA were used to measure the students’ skill levels. Katira et al. [121] noted that having two programmers from different genders is less likely to result in compatibility; whereas minority pairs are more likely to be perceived as compatible.
5.3.2 Applying ANP in Pair Programming

There is continued concern about matching two programmers, and several studies have pointed out various factors, such as experience, gender, programming style, and personality type, that should be considered when matching two programmers, in order to increase the quality and productivity of coding practice [122]. Therefore, in pair programming, ANP can be applied for two purposes:

1. Applying the ANP to select the best pair matching with respect to four criteria, which are sharing knowledge, speed, code quality, and learning. Poff [123] introduced four possible pairs as alternatives, which are (1) expert-expert pairing, (2) expert-average pairing, (3) expert-novice pairing, (4) novice-novice pairing.

2. Applying the ANP to rank the rules of matching pairs. In other words, this involves investigating the establishing of pairs with similar or different characteristics. The criteria for ranking the rules for matching are the same as the criteria mentioned previously, and the alternatives are (1) almost identical, (2) marginally different, and (3) clearly different.

5.3.3 First: Selecting the Best Matching Pairs

The first application of the ANP is to choose the best matching pairs with respect to the mentioned criteria. These criteria are explained in the following part.

Proposed Criteria for Selecting the Best Matching Pairs

To select the best matching pairs, it is important to identify the criteria that affect the selection process. These criteria are compared with each other to show the interdependencies, and also, compared with respect to each alternative or pair. The matching pairs are compared with respect to the criteria in order to show the feedback
relation in the selection process. In this thesis, four criteria are proposed for selecting the best matching pairs. These criteria are:

- **Sharing Knowledge**: Which pairs are preferable in term of maximizing knowledge exchange?

- **Learning**: Which pairs will be involved in this type of training and learning environment?

- **Speed**: Which pairs are preferable in terms of speeding up the coding phase?

- **Code Quality**: Which pairs are preferable in terms of enhancing the code quality?

The proposed alternatives are explained below. These pairs have been described in more detail by Williams and Kessler in [124].

- **Expert-Expert Pairing**: Matching two experts together can produce better code in less time. Each expert works with high confidence in his/her abilities. Each expert has expertise in a certain part of the code, which offers a strong likelihood of success [123]. However, the drawback with this type of matching is that large egos could be a concern [123]. Williams and Kessler [124] emphasize that, “when the two experts get in sync, you can hear the lightning crackling. Working with a good expert partner is like gaining 40 or more IQ points” [124].

- **Expert-Average Pairing**: Having an inspired average partner is a key factor to achieving success in this type of pairing. This pairing will not work well if the average has no interest in increasing his/her knowledge. However, the expert has to be aware that his/her partner is less experienced, which leads to slowing down from the normal pace [123]. The expert also has to put more effort into observing the average programmer's work.
Expert-Novice Pairing: This pairing will give the novice an opportunity to gain education by pairing with the expert. This will definitely minimize the expert’s productivity, since most of the work has to be described to the novice. Therefore, the expert should not be under pressure in terms of a tight deadline, and he/she should be patient and willing to teach the novice. This kind of teaching is advantageous to the expert, as in this pairing they are therefore also able to learn more about the topic [123].

Novice-Novice Pairing: The main advantage of this pairing is that both programmers will have a chance to learn faster than if they were alone [123]. A mentor will train them, then the pair works together to understand the instructions. Having a mentor is important in this collaboration, in order to ensure that the two novices are on track or to stop them from getting stuck in the project [123].

ANP Structure for Selecting the Best Pairs

Structuring the problem in a network is the first stage in the ANP. The network for selecting the best pairs includes three clusters. The objective cluster in the network selects the best pair. The second cluster is the criteria, which includes the following nodes: sharing knowledge, code quality, learning, and speed. The third cluster is alternatives, which includes the following pairings: expert-expert, expert-average, expert-novice, and novice-novice.

Figure 5.7 shows the ANP network for selecting the best matching pairs.

Pairwise Comparisons for Selecting the Best Pairs

The ANP team was asked to evaluate the importance of each criterion with respect to all of the matching pairs, in order to show the interdependencies. In addition, the team was asked to evaluate each matching pair with respect to all of the criteria,
in order to show the feedback of the network. The Team 1 participants received the suitable ANP tables, in order to make it easy for the participants to perform the pairwise comparisons based on the ANP fundamental scale that was described previously. For the interdependencies relationship, the participants were asked the following:

- With respect to Expert-Expert Pairing: which criterion is more important, sharing knowledge or speed and by how much?

- With respect to Expert-Expert Pairing: which criterion is more important, speed or learning and by how much?

- With respect to Expert-Expert Pairing: which criterion is more important, code quality or sharing knowledge and by how much?

- With respect to Expert-Expert Pairing: which criterion is more important, learning or code quality and by how much?

- With respect to Expert-Expert Pairing: which criterion is more important, learning or sharing knowledge and by how much?
• With respect to Expert-Expert Pairing: which criterion is more important, speed or code quality and by how much?

• With respect to Expert-Average Pairing: which criterion is more important, sharing knowledge or speed and by how much?

• With respect to Expert-Average Pairing: which criterion is more important, speed or learning and by how much?

• With respect to Expert-Average Pairing: which criterion is more important, code quality or sharing knowledge and by how much?

• With respect to Expert-Average Pairing: which criterion is more important, learning or code quality and by how much?

• With respect to Expert-Average Pairing: which criterion is more important, learning or sharing knowledge and by how much?

• With respect to Expert-Average Pairing: which criterion is more important, speed or code quality and by how much?

• With respect to Expert-Novice Pairing: which criterion is more important, sharing knowledge or speed and by how much?

• With respect to Expert-Novice Pairing: which criterion is more important, speed or learning and by how much?

• With respect to Expert-Novice Pairing: which criterion is more important, code quality or sharing knowledge and by how much?

• With respect to Expert-Novice Pairing: which criterion is more important, learning or code quality and by how much?
• With respect to Expert-Novice Pairing: which criterion is more important, learning or sharing knowledge and by how much?

• With respect to Expert-Novice Pairing: which criterion is more important, speed or code quality and by how much?

• With respect to Novice-Novice Pairing: which criterion is more important, sharing knowledge or speed and by how much?

• With respect to Novice-Novice Pairing: which criterion is more important, speed or learning and by how much?

• With respect to Novice-Novice Pairing: which criterion is more important, code quality or sharing knowledge and by how much?

• With respect to Novice-Novice Pairing: which criterion is more important, learning or code quality and by how much?

The participants then compared each matching pair with respect to each criterion in order to evaluate the feedback relationship. Examples of these questions for the participants are:

• With respect to sharing knowledge: which pair do you prefer, Expert-Expert Pairing or Expert-Average Pairing?

• With respect to sharing knowledge: which pair do you prefer, Expert-Expert Pairing or Expert-Novice Pairing?
• With respect to sharing knowledge: which pair do you prefer, Expert-Expert Pairing or Novice-Novice Pairing?

• With respect to sharing knowledge: which pair do you prefer, Expert-Average Pairing or Expert-Novice Pairing?

• With respect to sharing knowledge: which pair do you prefer, Expert-Average Pairing or Novice-Novice Pairing?

• With respect to sharing knowledge: which pair do you prefer, Expert-Novice Pairing or Novice-Novice Pairing?

• With respect to speed: which pair do you prefer, Expert-Expert Pairing or Expert-Average Pairing?

• With respect to speed: which pair do you prefer, Expert-Expert Pairing or Expert-Novice Pairing?

• With respect to speed: which pair do you prefer, Expert-Expert Pairing or Novice-Novice Pairing?

• With respect to speed: which pair do you prefer, Expert-Average Pairing or Expert-Novice Pairing?

• With respect to speed: which pair do you prefer, Expert-Average Pairing or Novice-Novice Pairing?

• With respect to speed: which pair do you prefer, Expert-Novice Pairing or Novice-Novice Pairing?

• With respect to learning: which pair do you prefer, Expert-Expert Pairing or Expert-Average Pairing?
• With respect to learning: which pair do you prefer, Expert-Expert Pairing or Expert-Novice Pairing?

• With respect to learning: which pair do you prefer, Expert-Expert Pairing or Novice-Novice Pairing?

• With respect to learning: which pair do you prefer, Expert-Average Pairing or Expert-Novice Pairing?

• With respect to learning: which pair do you prefer, Expert-Average Pairing or Novice-Novice Pairing?

• With respect to learning: which pair do you prefer, Expert-Novice Pairing or Novice-Novice Pairing?

• With respect to code quality: which pair do you prefer, Expert-Expert Pairing or Expert-Average Pairing?

• With respect to code quality: which pair do you prefer, Expert-Expert Pairing or Expert-Novice Pairing?

• With respect to code quality: which pair do you prefer, Expert-Expert Pairing or Novice-Novice Pairing?

• With respect to code quality: which pair do you prefer, Expert-Average Pairing or Expert-Novice Pairing?

• With respect to code quality: which pair do you prefer, Expert-Average Pairing or Novice-Novice Pairing?

• With respect to code quality: which pair do you prefer, Expert-Novice Pairing or Novice-Novice Pairing?
Selecting Pairs Results Based on ANP Evaluation

Team 1's results show that the expert-expert pairing was the highest ranked among the four alternatives. Expert-average pairing came second, followed by expert-novice pairing, then novice-novice pairing. Table 5.16 shows the relative weight of each one as a percentage. In addition, in using the software, we are able to examine the importance of each criterion based on all of the alternatives. The code quality criterion was ranked as the most important among the criteria, followed by learning, sharing knowledge, and speed. Figure 5.8 shows the criteria importance scores.

Table 5.16: Matching pairs ranking for Team 1

<table>
<thead>
<tr>
<th>Pairs</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert-Expert</td>
<td>66.38 %</td>
</tr>
<tr>
<td>Expert-Average</td>
<td>21.01 %</td>
</tr>
<tr>
<td>Expert-Novice</td>
<td>8.51 %</td>
</tr>
<tr>
<td>Novice-Novice</td>
<td>4.08 %</td>
</tr>
</tbody>
</table>

Figure 5.8: The importance of the criteria by Team 1

Team 2's ranking results show that expert-expert pairing came first, followed by expert-average, expert-novice, and novice-novice. Table 5.17 displays the matching pairs ranking by Team 2. In terms of the most important factor for matching pairs,
Team 2 gave code quality the top score. Table 5.18 shows the ranking of the criteria by Team 2.

**Table 5.17:** Matching pairs ranking by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Expert-Expert</td>
</tr>
<tr>
<td>2</td>
<td>Expert-Average</td>
</tr>
<tr>
<td>3</td>
<td>Expert-Novice</td>
</tr>
<tr>
<td>4</td>
<td>Novice-Novice</td>
</tr>
</tbody>
</table>

**Table 5.18:** The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Code Quality</td>
</tr>
<tr>
<td>2</td>
<td>Sharing Knowledge</td>
</tr>
<tr>
<td>3</td>
<td>Learning</td>
</tr>
<tr>
<td>4</td>
<td>Speed</td>
</tr>
</tbody>
</table>

**Observations**

1. With respect to all of the criteria, both teams ranked expert-expert pairing as the most preferable pair, followed by expert-average, expert-novice, and novice-novice.

2. Team 2 ranked code quality as the most important criterion.

3. Similar to Team 2, Team 1 considered code quality as the most important criterion.

4. Learning was ranked as the second most important criterion by Team 1, followed by sharing knowledge and speed.

5. Team 2 ranked sharing knowledge as the second most important criterion, followed by learning and speed.
6. Considering each criterion individually, Team 1 ranked expert-expert the highest with respect to the code quality, speed, learning, and sharing knowledge criteria. Table 5.19 shows the ranking for all pairs with respect to each criterion.

7. Considering each pairing type individually, Team 1 ranked the learning factor as the most important with respect to expert-average, expert-novice, and novice-novice. Team 1 ranked code quality as the most important criterion according to expert-expert pairing. Table 5.20 shows weight of each criterion with respect to each matching pair.

**Table 5.19:** Ranking pairs with respect to each criterion for Team 1

<table>
<thead>
<tr>
<th>Pairs</th>
<th>Code Quality</th>
<th>Pairs</th>
<th>Speed</th>
<th>Pairs</th>
<th>Sharing Knowledge</th>
<th>Pairs</th>
<th>Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert-Expert</td>
<td>65.16 %</td>
<td>Expert-Expert</td>
<td>70.86 %</td>
<td>Expert-Expert</td>
<td>67.50 %</td>
<td>Expert-Expert</td>
<td>64.38 %</td>
</tr>
<tr>
<td>Expert-Average</td>
<td>21.62 %</td>
<td>Expert-Average</td>
<td>18.39 %</td>
<td>Expert-Average</td>
<td>19.40 %</td>
<td>Expert-Average</td>
<td>23.02 %</td>
</tr>
<tr>
<td>Expert-Novice</td>
<td>9.04 %</td>
<td>Expert-Novice</td>
<td>7.50 %</td>
<td>Expert-Average</td>
<td>8.61 %</td>
<td>Expert-Novice</td>
<td>8.19 %</td>
</tr>
<tr>
<td>Novice-Novice</td>
<td>4.16 %</td>
<td>Novice-Novice</td>
<td>3.24 %</td>
<td>Novice-Novice</td>
<td>4.47 %</td>
<td>Novice-Novice</td>
<td>4.39 %</td>
</tr>
</tbody>
</table>

**Table 5.20:** Criteria weights with respect to each pair for Team 1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Quality</td>
<td>64.70 %</td>
<td>9.83 %</td>
<td>9.02 %</td>
<td>7.68 %</td>
</tr>
<tr>
<td>Sharing Knowledge</td>
<td>4.67 %</td>
<td>30.98 %</td>
<td>31.47 %</td>
<td>29.36 %</td>
</tr>
<tr>
<td>Speed</td>
<td>26.12 %</td>
<td>4.51 %</td>
<td>3.66 %</td>
<td>6.90 %</td>
</tr>
<tr>
<td>Learning</td>
<td>4.48 %</td>
<td>54.66 %</td>
<td>55.83 %</td>
<td>56.03 %</td>
</tr>
</tbody>
</table>
5.3.4 Second: Ranking the Rules for Matching Pairs

In this section, a ranking method is presented in order to assist the development team in selecting the best rules for matching pairs. The proposed criteria and the rules for matching pairs will be explained as follows.

**Proposed Criteria for Rules of Matching Pairs**

Several researchers have identified various human features that influence cooperation between two programmers. These features include the following:

- **Experience**: One of the most significant personal characteristics is experience. According to Cockburn *et al.* “many experienced programmers are very reluctant to program with another person. Some say their code is ‘personal,’ or that another person would only slow them down. Others say working with a partner will cause trouble coordinating work times or code versions”[122]. In addition, Domino *et al.*[125] state that “experience has a strong positive link to performance. Should experience be shown to be a significant factor in pair programming success, then the relative experience of developers can be used as a selection criterion for pair programming teams”[125]. Moreover, based on the experiment of Domino *et al.*[125] “it is interesting to note that the more highly experienced programmers using pair programming in a face-to-face work setting reported higher levels of individual satisfaction than less experienced developers”[125].

- **Gender**: There are several studies that exhibit that gender in pair programming might affect performance. For example, Werner *et al.*[126] conducted an experiment which proved that merging both men and women in matching pairs could increase their efforts. One reason might be the competitive environment between the two genders. This section, therefore, compares how having pairs
from the same gender or different genders may affect matching pairs.

- **Personality**: This reflects various features, like being friendly, open-minded, flexible, creative, and quiet. Personality, in addition, can have a significant effect on communication among pairs and on each programmer’s style of work [127].

- **Culture**: Merging two programmers with different cultural backgrounds can be good for building trust and communication among the team members [124].

- **Programming Style**: Followed rules during the coding stage. Cockburn *et al.* state, based on an experiment, that “several well-respected programmers prefer working in pairs, making it their preferred programming style” [122].

**ANP Structure for Ranking Rules of Matching Pairs**

The ANP network for making rules for pair matching consists of three clusters: The objective cluster ranks the rules of matching pairs; the second cluster reflects the criteria, including experience, gender, programming style, personality, and culture; the third cluster contains three options for pairing programmers. These attributes should be almost identical, marginally different, or clearly different. Thus, the following structure seeks to answer questions like, what is the best alternative to take? With respect to all of the criteria, what will be the best decision? With respect to each criterion, what will be the best decision? Figure [5.9] illustrates the ANP network for the problem.

**Pairwise Comparisons for the Prioritization Methods**

The ANP team was asked to evaluate the importance of each criterion with respect to all of the pairing options, in order to show the interdependencies. In addition, the team was asked to evaluate each type of pairing with respect to all of the criteria,
in order to show the feedback of the network. The Team 1 participants received
the suitable ANP tables, in order to make it easy for the participants to perform
the pairwise comparisons based on the ANP fundamental scale that was described
previously. For the interdependencies relationship, the participants were asked the
following:

- With respect to Almost Identical: which criterion is more important, experience
or gender and by how much?

- With respect to Almost Identical: which criterion is more important, experience
or culture and by how much?

- With respect to Almost Identical: which criterion is more important, gender or
personality and by how much?

- With respect to Almost Identical: which criterion is more important, gender or
culture and by how much?

- With respect to Almost Identical: which criterion is more important, programming
style or experience and by how much?
• With respect to Almost Identical: which criterion is more important, programming style or culture and by how much?

• With respect to Almost Identical: which criterion is more important, personality or programming style and by how much?

• With respect to Almost Identical: which criterion is more important, personality or culture and by how much?

• With respect to Almost Identical: which criterion is more important, personality or experience and by how much?

• With respect to Almost Identical: which criterion is more important, gender or programming style and by how much?

• With respect to Marginally Different: which criterion is more important, experience or gender and by how much?

• With respect to Marginally Different: which criterion is more important, experience or culture and by how much?

• With respect to Marginally Different: which criterion is more important, gender or personality and by how much?

• With respect to Marginally Different: which criterion is more important, gender or culture and by how much?

• With respect to Marginally Different: which criterion is more important, programming style or experience and by how much?

• With respect to Marginally Different: which criterion is more important, programming style or culture and by how much?
• With respect to Marginally Different: which criterion is more important, personality or programming style and by how much?

• With respect to Marginally Different: which criterion is more important, personality or culture and by how much?

• With respect to Marginally Different: which criterion is more important, personality or experience and by how much?

• With respect to Marginally Different: which criterion is more important, gender or programming style and by how much?

• With respect to Clearly Different: which criterion is more important, experience or gender and by how much?

• With respect to Clearly Different: which criterion is more important, experience or culture and by how much?

• With respect to Clearly Different: which criterion is more important, gender or personality and by how much?

• With respect to Clearly Different: which criterion is more important, gender or culture and by how much?

• With respect to Clearly Different: which criterion is more important, programming style or experience and by how much?

• With respect to Clearly Different: which criterion is more important, programming style or culture and by how much?

• With respect to Clearly Different: which criterion is more important, personality or programming style and by how much?
• With respect to Clearly Different: which criterion is more important, personality or culture and by how much?

• With respect to Clearly Different: which criterion is more important, personality or experience and by how much?

• With respect to Clearly Different: which criterion is more important, gender or programming style and by how much?

The participants then compared the three options pairs with respect to each criterion, in order to evaluate the feedback relationship. Examples of these questions for the participants are:

• With respect to experience: which pair do you prefer, Almost Identical or Marginally Different?

• With respect to experience: which pair do you prefer, Almost Identical or Clearly Different?

• With respect to experience: which pair do you prefer, Marginally Different or Clearly Different?

• With respect to gender: which pair do you prefer, Almost Identical or Marginally Different?

• With respect to gender: which pair do you prefer, Almost Identical or Clearly Different?

• With respect to gender: which pair do you prefer, Marginally Different or Clearly Different?

• With respect to programming style: which pair do you prefer, Almost Identical or Marginally Different?
• With respect to programming style: which pair do you prefer, Almost Identical or Clearly Different?

• With respect to programming style: which pair do you prefer, Marginally Different or Clearly Different?

• With respect to personality: which pair do you prefer, Almost Identical or Marginally Different?

• With respect to personality: which pair do you prefer, Almost Identical or Clearly Different?

• With respect to personality: which pair do you prefer, Marginally Different or Clearly Different?

• With respect to culture: which pair do you prefer, Almost Identical or Marginally Different?

• With respect to culture: which pair do you prefer, Almost Identical or Clearly Different?

• With respect to culture: which pair do you prefer, Marginally Different or Clearly Different?

Rules of Matching Pairs Results Based on ANP Evaluation

Team 1’s results show that an almost identical pair was given the highest rank among the three alternatives. Marginally different pair came second, followed by clearly different pair. Table 5.21 shows the relative weight of each one as a percentage. In addition, in using the software, we are able to examine the importance of each criterion based on all of the alternatives. Personality was ranked as the most important among the criteria, followed by gender, culture, experience, and programming style. Figure 5.10 shows the criteria importance scores.
Table 5.21: Matching pairs ranking for Team 1

<table>
<thead>
<tr>
<th>Rules in Pairing</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Almost Identical</td>
<td>61.69 %</td>
</tr>
<tr>
<td>Marginally Different</td>
<td>20.25 %</td>
</tr>
<tr>
<td>Clearly Different</td>
<td>18.05 %</td>
</tr>
</tbody>
</table>

Figure 5.10: The importance of the criteria by Team 1 in the rules of matching

Similar to Team 1, Team 2’s results show that the team decided to rank almost identical pair as the highest alternative. Table 5.22 displays the rules of pairing ranking by Team 2. In terms of the most important factor in the rules of pairing, Team 2 gave experience the top score. Table 5.23 shows the ranking of the criteria by Team 2.

Table 5.22: Rules of pairing ranking by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Rules In Pairing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Almost Identical</td>
</tr>
<tr>
<td>2</td>
<td>Marginally Different</td>
</tr>
<tr>
<td>3</td>
<td>Clearly Different</td>
</tr>
</tbody>
</table>
Table 5.23: The importance of the criteria by Team 2 in the rules of pairing

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Experience</td>
</tr>
<tr>
<td>2</td>
<td>Personality</td>
</tr>
<tr>
<td>3</td>
<td>Programming Style</td>
</tr>
<tr>
<td>4</td>
<td>Gender</td>
</tr>
<tr>
<td>5</td>
<td>Culture</td>
</tr>
</tbody>
</table>

Observations

1. With respect to all of the criteria, Team 1 ranked the almost identical option as the highest alternative. Also, Team 1 ranked the marginally different option in the second position, followed by clearly different in the last position.

2. Team 2 ranked almost identical as the highest option based on the traditional method of XP. Also, in terms of the most important criteria, the Team 2 members ranked experience as the most important, while culture was a less important factor.

3. Considering each criterion individually, we can notice that the almost identical option was given the top score by Team 1 in terms of experience, programming style, and personality. The clearly different option was ranked the highest with respect to gender. Table 5.24 shows the ranking for all pairing options with respect to each criterion.

4. Considering each pairing option individually, we can notice that personality was given the highest importance by Team 1 with respect to the three options. Table 5.25 shows the weight of each criterion with respect to each prioritization technique.

5. Regarding the most preferable pairing option in terms of gender, Team 2 ranked clearly different as the highest.
**Table 5.24:** Pairing options with respect to each criterion individually for Team 1

<table>
<thead>
<tr>
<th>Rules in Pairing</th>
<th>Experience</th>
<th>Rules in Pairing</th>
<th>Gender</th>
<th>Rules in Pairing</th>
<th>Programming Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>Almost-Identical</td>
<td>73.06 %</td>
<td>Clearly-Different</td>
<td>64.91 %</td>
<td>Almost-Identical</td>
<td>77.20 %</td>
</tr>
<tr>
<td>Marginally Different</td>
<td>18.83 %</td>
<td>Marginally Different</td>
<td>27.89 %</td>
<td>Marginally Different</td>
<td>17.34 %</td>
</tr>
<tr>
<td>Clearly-Different</td>
<td>8.09 %</td>
<td>Almost-Identical</td>
<td>7.19 %</td>
<td>Clearly-Different</td>
<td>5.45 %</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rules in Pairing</th>
<th>Personality</th>
<th>Rules in Pairing</th>
<th>Culture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Almost Identical</td>
<td>75.14 %</td>
<td>Almost Identical</td>
<td>71.47 %</td>
</tr>
<tr>
<td>Marginally Different</td>
<td>17.81 %</td>
<td>Marginally Different</td>
<td>21.84 %</td>
</tr>
<tr>
<td>Clearly Different</td>
<td>7.04</td>
<td>Clearly Different</td>
<td>6.68 %</td>
</tr>
</tbody>
</table>

**Table 5.25:** Criteria weights with respect to each pairing option for Team 1

<table>
<thead>
<tr>
<th></th>
<th>Almost Identical</th>
<th>Marginally Different</th>
<th>Clearly Different</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experience</td>
<td>5.49 %</td>
<td>8.63 %</td>
<td>5.23 %</td>
</tr>
<tr>
<td>Gender</td>
<td>19.74 %</td>
<td>23.55 %</td>
<td>11.74 %</td>
</tr>
<tr>
<td>Programming Style</td>
<td>2.53 %</td>
<td>4.37 %</td>
<td>3.44 %</td>
</tr>
<tr>
<td>Personality</td>
<td>62.67 %</td>
<td>51.35 %</td>
<td>61.34 %</td>
</tr>
<tr>
<td>Culture</td>
<td>9.53 %</td>
<td>12.07 %</td>
<td>18.23 %</td>
</tr>
</tbody>
</table>
5.4 ANP in Refactoring

In this chapter, two applications of ANP in refactoring practice are presented. This section introduces the refactoring practice and recent related research. The ANP network structure that contains code quality attributes such as criteria, and refactoring techniques as alternatives to be ranked, are also shown in this chapter. The end of this section shows and discusses the findings of the case study and its results.

5.4.1 Introduction

The process of enhancing the structure of an existing code by altering the internal design without changing the external design is called code refactoring [128]. It is a significant issue in the XP development cycle to enhance software design, and to minimize the cost and effort that are needed for testing and coding.

Some researchers have concentrated on guidelines for the refactoring process; for example, a 3-stage model has been introduced by Kataoka et al. [129], and the model contains “identification of refactoring candidates, validation of refactoring effects, and application of refactoring” [129]. Meanwhile, Mens and Tourwe [130] have described the refactoring phases in more detail. These phases start with specifying the portion of the software that should be refactored, then determining which refactoring technique is suitable to be applied, performing the refactoring, and finally measuring the influence of the applied refactoring technique on the code quality [130].

Other researchers investigated various aspects of refactoring techniques. Simmonds and Mens [131] studied four software-refactoring methods: Eclipse, Together ControlCenter 6.0, SmalltalkWorks 7.0, and Gurn. In addition, Murphy-Hill et al. [132] conducted an empirical investigation to compare four techniques. These four techniques were applied to collect refactoring data in order to assist in establishing a powerful refactoring technique.
Maticorna and Perez [133] introduced refactoring interpretation and the possibility of using it as a method in order to compare various refactoring explanations, involving refactoring catalogs. Moreover, Maticorna and Perez [133] have worked on various refactoring concerns, like actions, application on scheduling, design, and scope, which might lead to the building of refactoring tools.

Several open-source Java systems have been investigated by Brunel et al. [134] in order to measure the accuracy of refactoring methods. This measuring is done by examining the following Java systems: MegaMek, Velocity, Antlr, HSQLDB, PDF-Box, Tyrant, and JasperReports. Murphy-Hill [135] built a model to investigate how refactoring techniques work in terms of the style of the refactoring browser. The model is made up of the following phases: identify, initiate, and execute [135].

Roberts et al. [136] examined the practical factors and technical requirements for the refactoring techniques. The authors emphasized that the ability to search the whole program and the accuracy are the most technical requirements. In addition, integration and speed are the most practical factors.

Marija and Kresimir [137] evaluated seven refactoring tools in order to choose the most suitable one. The seven tools were: Refactoring Browser (Smalltalk), Eclipse (C++, Java), Refactor (C# VB.NET, C++, ASP.NET), IntelliJ Idea (Java), Refactor (C# VB.NET, ASP.NET), NDepend (.NET code base), and Refactor (C++, Java). These refactoring tools were compared to each other concerning various issues, such as reliability, scalability, automation, discover-ability, coverage, and configurability.

Mahmood and Reddy [138] examined three refactoring techniques in order to avoid human errors while performing the manual refactoring. The authors evaluated the following refactoring tools: JBuilder 2008, RefactorIT 2.7 beta, and IntelliJ Idea 7.0.4. The authors compared these techniques with respect to various issues, such as user control, consistency, information processing, user experience, goal assessment, errors, design for the user, and ease of use. The authors proposed some enhancements...
in order to maximize the consistency of software usability.

Other studies focused on the identification of code smells in order to locate possible refactoring. For example, Hayashi et al. [139] introduced a tool for Eclipse using plugins. This tool directs the developer in terms of how to perform refactoring and which part of the code uses the histories of program modification. The proposed tool focused on answering the following questions: Where to refactor? Which suitable refactoring technique should be used? When should refactoring be applied?

5.4.2 Refactoring Techniques

Refactoring assists the development team to enhance the software design, understand the software more easily, find errors, and program faster, as Fowler et al. [128] confirmed. Fowler et al. [128] specified several refactoring techniques and arranged them into the following categories: moving features between objects, making method calls simpler, simplifying conditional expressions, composing methods, dealing with generalization, and organizing data. Each of these categories influences the quality attributes. Each project might have different quality attribute priorities, and using the refactoring techniques enhances the software design and the code. Therefore, in order to maximize the benefit from the system, it is important to assign the developers’ efforts to the most significant quality attributes. Selecting the refactoring techniques consumes time and might lead to conflicting opinions.

In this section, the main objective is to rank refactoring techniques according to their influence on the code quality. Five refactoring techniques have been selected in this study, in order to examine their importance using the ANP. These techniques were selected from the four different groups introduced by Fowler et al. [128]. The selected refactoring techniques are: Extract Method, Extract Class, Inline Class, Pull UP Method, and Rename Method.
5.4.3 Applying ANP in Refactoring

The main objective of applying the ANP in refactoring is to assist the XP team members in ranking the refactoring techniques with respect to the code quality attributes. In this section, there are two uses of the ANP in refactoring. The first use is ranking the refactoring techniques based on internal quality attributes, and the second use is to rank the refactoring techniques based on external quality attributes.

5.4.4 First: Ranking Refactoring Methods Based on Internal Quality Attributes

This section will introduce some previous studies that have examined the effect of refactoring techniques on the internal code quality attributes. This is following by introducing the ANP applying to rank the refactoring techniques.

Zhao and Hayes [140] conducted two case studies in order to investigate an approach that specifies which packages and classes need to be refactored according to various measures, like complexity, coupling, and code size. Using a measure-driven refactoring decision, the authors presented a rank-based software in order to support the team members’ decisions about where resources can be applied during refactoring.

Dallal and Briand [141] presented an automated refactoring method to enhance the cohesion of the software in order to enhance program testability. Sahraoui et al. [142] organized an empirical study in order to examine the effect of coupling and inheritance metrics on maintainability. The authors discovered a portion of the system that needed to be enhanced and refactored.

Stroulija and Kapoor [143] studied the possibility of enhancing the design and code quality using refactoring. Several refactoring techniques, such as Extract Abstract Class and Extract Superclass, were applied, and the results showed decreases in the number of methods, the number of statements, lines of code, and the number of
collaborators in the individual system classes.

Moser et al. [144] organized a case study to investigate the influence of refactoring on the internal quality attributes of source code. The case study was done in an Agile environment, and the selected quality attributes were coupling, response of class, number of children, number of methods per class, depth of inheritance tree, cohesion, and complexity. Based on their proposed method, the authors found that refactoring might enhance the internal metrics of object-oriented classes that are written in Java for reusability.

Bois and Mens [145] introduced a framework for the internal code qualities, like cohesion, number of children, number of methods, coupling, and response for a class. In order to achieve this, the authors investigated various refactoring techniques, such as Encapsulate Filled, Pull Up Method, and Extract Method.

Elish and Alshayeb [146] categorized refactoring patterns according to their influence on external and internal code quality attributes. The authors selected the following refactoring techniques: Form Template Method, Replace Construction with Creation Methods, Replace Conditional Dispatcher with Command, Chain Constructors, Introduce Null Object, Unify Interface, and Compose Method. The authors investigated different internal code quality metrics such as Number of Test Cases (NOTC) for the size of test case, Lines of Code for Class (LOCC), FOUT, LOC, DIT, LCOM, Number of Methods (NOM), Number of Fields (NOF), Number of Children (NOC), RFC, and WMC.

Over the course of 15 months, Ratzinger et al. [147] evaluated an industrial system. The authors exhibited the way that refactoring could improve the software evolvability and minimize the change couplings. In addition, Kataoka et al. [148] emphasized that refactoring patterns like extract class and extract method enhance system maintainability and minimize coupling in the code.
Proposed Criteria for Ranking the Refactoring Techniques

It is important to specify code quality attributes in order to rank refactoring techniques. The code quality attributes should be identified based on their value to the organization or the team member. Different projects will have different factors and alternative refactoring techniques to be examined. In this thesis, there are four internal code quality attributes that are selected as criteria used to rank the refactoring patterns:

- **Complexity**: The degree of connectivity among components of a design unit [149].

- **Cohesion**: Each component implements one function and implements it well [149].

- **Code Size**: Size in terms of number of files, number of lines of code (#LOC), functions, tables, classes, etc [149].

- **Coupling**: The strength of the interconnections between the system components [149].

ANP Structure for Ranking Refactoring Methods Based on the Internal Attributes

Structuring the problem as a network that consists of three clusters is the first step in the ANP. The first cluster is the objective, which ranks the refactoring patterns. The second cluster contains the criteria: coupling, code size, complexity, and cohesion. The third cluster includes the alternatives: Pull Up Method, Extract Class, Rename Method, Extract Method, and Inline Class. Figure 5.11 shows the ANP structure for the problem.
Pairwise Comparisons for the Refactoring Techniques

The participants have applied the refactoring techniques in their XP project in order to note the effect on the code. After that, based on the proposed criteria, the students evaluated each refactoring technique. The ANP team received the suitable ANP papers and tables in order to facilitate the comparisons process. For the interdependencies relationship, the participants were asked the following:

- With respect to Extract Method: which criterion is more important, cohesion or coupling and by how much?
- With respect to Extract Method: which criterion is more important, coupling or code size and by how much?
- With respect to Extract Method: which criterion is more important, complexity or cohesion and by how much?
- With respect to Extract Method: which criterion is more important, code size or complexity and by how much?
• With respect to Extract Method: which criterion is more important, code size or cohesion and by how much?

• With respect to Extract Method: which criterion is more important, coupling or complexity and by how much? item With respect to Extract Class: which criterion is more important, cohesion or coupling and by how much?

• With respect to Extract Class: which criterion is more important, coupling or code size and by how much?

• With respect to Extract Class: which criterion is more important, complexity or cohesion and by how much?

• With respect to Extract Class: which criterion is more important, code size or complexity and by how much?

• With respect to Extract Class: which criterion is more important, code size or cohesion and by how much?

• With respect to Extract Class: which criterion is more important, coupling or complexity and by how much? item With respect to Inline Class: which criterion is more important, cohesion or coupling and by how much?

• With respect to Inline Class: which criterion is more important, coupling or code size and by how much?

• With respect to Inline Class: which criterion is more important, complexity or cohesion and by how much?

• With respect to Inline Class: which criterion is more important, code size or complexity and by how much?

• With respect to Inline Class: which criterion is more important, code size or cohesion and by how much?
• With respect to Inline Class: which criterion is more important, coupling or complexity and by how much? item

• With respect to Pull Up Method: which criterion is more important, cohesion or coupling and by how much?

• With respect to Pull Up Method: which criterion is more important, coupling or code size and by how much?

• With respect to Pull Up Method: which criterion is more important, complexity or cohesion and by how much?

• With respect to Pull Up Method: which criterion is more important, code size or complexity and by how much?

• With respect to Pull Up Method: which criterion is more important, code size or cohesion and by how much?

• With respect to Pull Up Method: which criterion is more important, coupling or complexity and by how much? item

• With respect to Rename Method: which criterion is more important, cohesion or coupling and by how much?

• With respect to Rename Method: which criterion is more important, coupling or code size and by how much?

• With respect to Rename Method: which criterion is more important, complexity or cohesion and by how much?

• With respect to Rename Method: which criterion is more important, code size or complexity and by how much?

• With respect to Rename Method: which criterion is more important, code size or cohesion and by how much?

• With respect to Rename Method: which criterion is more important, coupling or complexity and by how much?
The participants then compared the refactoring patterns with respect to each criterion in order to evaluate the feedback relationship. Examples of these questions for the participants are:

- With respect to cohesion: which method do you prefer, Extract Method or Extract Class?
- With respect to cohesion: which method do you prefer, Extract Method or Inline Class?
- With respect to cohesion: which method do you prefer, Extract Method or Pull Up Method?
- With respect to cohesion: which method do you prefer, Extract Method or Rename Method?
- With respect to cohesion: which method do you prefer, Extract Class or Inline Class?
- With respect to cohesion: which method do you prefer, Extract Class or Pull Up Method?
- With respect to cohesion: which method do you prefer, Extract Class or Rename Method?
- With respect to cohesion: which method do you prefer, Inline Class or Pull Up Method?
- With respect to cohesion: which method do you prefer, Inline Class or Rename Method?
- With respect to cohesion: which method do you prefer, Pull Up Method or Rename Method?
• With respect to coupling: which method do you prefer, Extract Method or Extract Class?

• With respect to coupling: which method do you prefer, Extract Method or Inline Class?

• With respect to coupling: which method do you prefer, Extract Method or Pull Up Method?

• With respect to coupling: which method do you prefer, Extract Method or Rename Method?

• With respect to coupling: which method do you prefer, Extract Class or Inline Class?

• With respect to coupling: which method do you prefer, Extract Class or Pull Up Method?

• With respect to coupling: which method do you prefer, Extract Class or Rename Method?

• With respect to coupling: which method do you prefer, Inline Class or Pull Up Method?

• With respect to coupling: which method do you prefer, Inline Class or Rename Method?

• With respect to coupling: which method do you prefer, Pull Up Method or Rename Method?

• With respect to complexity: which method do you prefer, Extract Method or Extract Class?
• With respect to complexity: which method do you prefer, Extract Method or Inline Class?

• With respect to complexity: which method do you prefer, Extract Method or Pull Up Method?

• With respect to complexity: which method do you prefer, Extract Method or Rename Method?

• With respect to complexity: which method do you prefer, Extract Class or Inline Class?

• With respect to complexity: which method do you prefer, Extract Class or Pull Up Method?

• With respect to complexity: which method do you prefer, Extract Class or Rename Method?

• With respect to complexity: which method do you prefer, Inline Class or Pull Up Method?

• With respect to complexity: which method do you prefer, Inline Class or Rename Method?

• With respect to complexity: which method do you prefer, Pull Up Method or Rename Method?

• With respect to code size: which method do you prefer, Extract Method or Extract Class?

• With respect to code size: which method do you prefer, Extract Method or Inline Class?
• With respect to code size: which method do you prefer, Extract Method or Pull Up Method?

• With respect to code size: which method do you prefer, Extract Method or Rename Method?

• With respect to code size: which method do you prefer, Extract Class or Inline Class?

• With respect to code size: which method do you prefer, Extract Class or Pull Up Method?

• With respect to code size: which method do you prefer, Extract Class or Rename Method?

• With respect to code size: which method do you prefer, Inline Class or Pull Up Method?

• With respect to code size: which method do you prefer, Inline Class or Rename Method?

• With respect to code size: which method do you prefer, Pull Up Method or Rename Method?

**Refactoring Techniques Results Based on ANP Evaluation**

Team 1’s results of ranking the refactoring patterns with respect to all four criteria is as follows: first, Extract Method; second, Extract Class; third, Pull Up Method; fourth, Inline Class; and fifth, Rename Method. Table 5.26 shows the scores of each pattern. Team 1 ranked cohesion as the most important criterion, followed by complexity in the second position, while code size and coupling were ranked in the third and fourth positions, respectively. Figure 5.12 exhibits the importance of each criterion as a percentage according to Team 1.
Table 5.26: Ranking the refactoring patterns based on internal attributes by Team 1

<table>
<thead>
<tr>
<th>Refactoring Patterns</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Method</td>
<td>29.17 %</td>
</tr>
<tr>
<td>Extract Class</td>
<td>25.01 %</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>18.74 %</td>
</tr>
<tr>
<td>Inline Class</td>
<td>17.58 %</td>
</tr>
<tr>
<td>Rename Method</td>
<td>9.48 %</td>
</tr>
</tbody>
</table>

Team 2 ranked the refactoring patterns as follows: first, Extract Class; second, Extract Method; third, Inline Class; fourth, Pull Up Method; and fifth, Rename Method. Table 5.27 displays the ranking of refactoring patterns by Team 2. Moreover, in terms of the most important criterion, Team 2 ranked coupling in the first position. Table 5.28 shows the ranking of the criteria by Team 2.

Table 5.27: Refactoring techniques ranking by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Refactoring Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Extract Class</td>
</tr>
<tr>
<td>2</td>
<td>Extract Method</td>
</tr>
<tr>
<td>3</td>
<td>Inline Class</td>
</tr>
<tr>
<td>4</td>
<td>Pull Up Method</td>
</tr>
<tr>
<td>5</td>
<td>Rename Method</td>
</tr>
</tbody>
</table>

Figure 5.12: The importance of the internal attributes for the refactoring patterns by Team 1
Table 5.28: The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Coupling</td>
</tr>
<tr>
<td>2</td>
<td>Cohesion</td>
</tr>
<tr>
<td>3</td>
<td>Complexity</td>
</tr>
<tr>
<td>4</td>
<td>Code Size</td>
</tr>
</tbody>
</table>

Observations

1. With respect to all of the criteria, Team 1 ranked the Extract Method as the highest refactoring technique.

2. Team 2 ranked Extract Class as the highest refactoring technique.

3. Both teams ranked Rename Method in the last position.

4. Team 1 ranked cohesion as the most important criterion, while Team 2 ranked coupling as the most important criterion.

5. With respect to each criterion individually, Team 1 ranked the Pull Up Method highest in terms of code size. Table 5.29 shows the ranking of all refactoring techniques with respect to each criterion.

6. Inline Class was ranked highest with respect to the coupling criterion by Team 1.

7. With respect to each refactoring technique individually, we can see that reducing complexity was ranked highest according to Rename Method. Table 5.30 shows the weight of each criterion with respect to each refactoring pattern.
Table 5.29: Refactoring techniques with respect to each criterion for Team 1

<table>
<thead>
<tr>
<th>Techniques</th>
<th>Cohesion</th>
<th>Techniques</th>
<th>Coupling</th>
<th>Techniques</th>
<th>Code-Size</th>
<th>Techniques</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Method</td>
<td>53.09 %</td>
<td>Inline Class</td>
<td>46.23 %</td>
<td>Pull Up Method</td>
<td>41.22 %</td>
<td>Extract Class</td>
<td>47.30 %</td>
</tr>
<tr>
<td>Extract Class</td>
<td>28.42 %</td>
<td>Pull Up Method</td>
<td>28.51 %</td>
<td>Inline Class</td>
<td>31.20 %</td>
<td>Extract Method</td>
<td>23.16 %</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>9.26 %</td>
<td>Rename Method</td>
<td>11.96 %</td>
<td>Rename Method</td>
<td>13.44 %</td>
<td>Inline Class</td>
<td>11.38 %</td>
</tr>
<tr>
<td>Rename Method</td>
<td>5.91 %</td>
<td>Extract Method</td>
<td>7.75 %</td>
<td>Extract Method</td>
<td>7.96 %</td>
<td>Rename Method</td>
<td>10.10 %</td>
</tr>
<tr>
<td>Inline Class</td>
<td>3.29 %</td>
<td>Extract Class</td>
<td>5.52 %</td>
<td>Extract Class</td>
<td>6.16 %</td>
<td>Pull Up Method</td>
<td>8.04 %</td>
</tr>
</tbody>
</table>

Table 5.30: Criteria weights with respect to each refactoring technique for Team 1

<table>
<thead>
<tr>
<th></th>
<th>Extract Method</th>
<th>Extract Class</th>
<th>Inline Class</th>
<th>Pull Up Method</th>
<th>Rename Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cohesion</td>
<td>64.32 %</td>
<td>66.36 %</td>
<td>7.01 %</td>
<td>5.48 %</td>
<td>15.54 %</td>
</tr>
<tr>
<td>Coupling</td>
<td>11.32 %</td>
<td>10.36 %</td>
<td>14.13 %</td>
<td>26.61 %</td>
<td>6.29 %</td>
</tr>
<tr>
<td>Code-Size</td>
<td>20.28 %</td>
<td>18.24 %</td>
<td>22.33 %</td>
<td>34.88 %</td>
<td>6.29 %</td>
</tr>
<tr>
<td>Complexity</td>
<td>4.06 %</td>
<td>5.02 %</td>
<td>56.51 %</td>
<td>33.01 %</td>
<td>71.86 %</td>
</tr>
</tbody>
</table>
5.4.5 Second: Ranking Refactoring Methods Based on External Quality Attributes

This section introduces the effect of refactoring techniques on external quality attributes before applying the ANP to refactoring practice.

Elish and Alshayeb [146], [150] identified refactoring patterns according to their influence on the external and internal code quality attributes. The authors selected the following external code quality attributes: reusability, adaptability, testability, completeness, understandability, and maintainability.

Using a dependency investigation, Leitch and Stroulia [151] examined the effects of refactoring patterns on software costs, efforts, and maintenance. Meanwhile, Kataoka et al. [152] introduced a quantitative analysis approach in order to assess maintainability improvements due to refactoring. The focus was on coupling attributes in order to measure the impact of refactoring.

Stroulia and Leitch [153] introduced an approach to evaluate the predicted cost of software maintenance by anticipating the return on investment (ROI) for the refactoring practice. Stroulia and Leitch [153] believe that this could enhance the code quality attributes, such as maintainability and performance, and could also maximize refactoring activities adoption.

Raed and Li [154] applied the hierarchy quality model to examine the impact of refactoring patterns on various metrics, such as effectiveness, flexibility, reusability, and extendibility. The authors concluded that not all refactoring patterns enhance code quality attributes.

Alshayeb [154] focused on validating or invalidating the impacts of refactoring on the following external attributes: understandability, adaptability, reusability, testability, and maintainability. The main objective of the study was to decide if it is beneficial to practice refactoring or not. The results showed that refactoring does not necessarily enhance the studied code quality attributes.
Proposed Criteria for Ranking the Refactoring Techniques

Similar to the internal code attributes, it is important to specify the most desirable external code quality attributes in order to rank the refactoring techniques. The external code quality attributes should be identified based on their value to the organization or the team member. Different projects will have different factors and alternative refactoring techniques to be examined. In this thesis, there are four external code quality attributes that are selected as criteria in order to rank the refactoring patterns:

- **Flexibility**: The ability of the system to adapt to different conditions and environments. Furthermore, this involves the ability to be managed during various changes [155]. Meier et al. [155] stated that, “a flexible system is one that is easy to reconfigure or adapt in response to different user and system requirements” [155].

- **Understandability**: The clarity of the meaning of each system component to a user [156].

- **Reusability**: The ability of subsystems and components to be appropriate for use in more than one application [156, 157].

- **Maintainability**: The ability of the software to easily accept changes, such as enhancing performance, adapting to new environments, or adjusting a component to fix faults [158].

ANP Structure for Ranking Refactoring Methods Based on the External Attributes

The main cluster ranks the refactoring patterns; the second cluster contains the criteria, which are flexibility, understandability, reusability, and maintainability; the
third cluster consists of the alternatives, which are Extract Class, Extract Method, Inline Class, Pull Up Method, and Rename Method. Figure 5.13 illustrates the ANP structure for ranking the refactoring techniques based on the external attributes.

**Figure 5.13:** ANP network for ranking refactoring techniques based on the external attributes

**Pairwise Comparisons for the Refactoring Techniques**

Similar to the internal attributes, the participants applied the refactoring techniques in their XP project in order to note the effect on the code. After that, based on the proposed criteria, the students evaluated each refactoring technique. The ANP team received the suitable ANP papers and tables in order to facilitate the comparisons process. For the interdependencies relationship, the participants were asked the following:

- With respect to Extract Method: which criterion is more important, flexibility or understandability and by how much?

- With respect to Extract Method: which criterion is more important, understandability or reusability and by how much?
• With respect to Extract Method: which criterion is more important, maintainability or flexibility and by how much?

• With respect to Extract Method: which criterion is more important, reusability or maintainability and by how much?

• With respect to Extract Method: which criterion is more important, reusability or flexibility and by how much?

• With respect to Extract Method: which criterion is more important, understandability or maintainability and by how much? item With respect to Extract Class: which criterion is more important, flexibility or understandability and by how much?

• With respect to Extract Class: which criterion is more important, understandability or reusability and by how much?

• With respect to Extract Class: which criterion is more important, maintainability or understandability and by how much?

• With respect to Extract Class: which criterion is more important, maintainability or flexibility and by how much?

• With respect to Extract Class: which criterion is more important, reusability or maintainability and by how much?

• With respect to Extract Class: which criterion is more important, reusability or flexibility and by how much?

• With respect to Extract Class: which criterion is more important, understandability or maintainability and by how much? item With respect to Inline Class: which criterion is more important, flexibility or understandability and by how much?

• With respect to Inline Class: which criterion is more important, understandability or reusability and by how much?
• With respect to Inline Class: which criterion is more important, maintainability or flexibility and by how much?

• With respect to Inline Class: which criterion is more important, reusability or maintainability and by how much?

• With respect to Inline Class: which criterion is more important, reusability or flexibility and by how much?

• With respect to Inline Class: which criterion is more important, understandability or maintainability and by how much? item With respect to Pull Up Method: which criterion is more important, flexibility or understandability and by how much?

• With respect to Pull Up Method: which criterion is more important, understandability or reusability and by how much?

• With respect to Pull Up Method: which criterion is more important, maintainability or flexibility and by how much?

• With respect to Pull Up Method: which criterion is more important, reusability or maintainability and by how much?

• With respect to Pull Up Method: which criterion is more important, reusability or flexibility and by how much?

• With respect to Pull Up Method: which criterion is more important, understandability or maintainability and by how much? item With respect to Rename Method: which criterion is more important, flexibility or understandability and by how much?

• With respect to Rename Method: which criterion is more important, understandability or code size and by how much?
• With respect to Rename Method: which criterion is more important, maintainability or flexibility and by how much?

• With respect to Rename Method: which criterion is more important, reusability or maintainability and by how much?

• With respect to Rename Method: which criterion is more important, reusability or flexibility and by how much?

• With respect to Rename Method: which criterion is more important, understandability or maintainability and by how much?

The participants then compared the refactoring patterns with respect to each criterion in order to evaluate the feedback relationship. Examples of these questions for the participants are:

• With respect to flexibility: which method do you prefer, Extract Method or Extract Class?

• With respect to flexibility: which method do you prefer, Extract Method or Inline Class?

• With respect to flexibility: which method do you prefer, Extract Method or Pull Up Method?

• With respect to flexibility: which method do you prefer, Extract Method or Rename Method?

• With respect to flexibility: which method do you prefer, Extract Class or Inline Class?

• With respect to flexibility: which method do you prefer, Extract Class or Pull Up Method?
• With respect to flexibility: which method do you prefer, Extract Class or Rename Method?

• With respect to flexibility: which method do you prefer, Inline Class or Pull Up Method?

• With respect to flexibility: which method do you prefer, Inline Class or Rename Method?

• With respect to flexibility: which method do you prefer, Pull Up Method or Rename Method?

• With respect to understandability: which method do you prefer, Extract Method or Extract Class?

• With respect to understandability: which method do you prefer, Extract Method or Inline Class?

• With respect to understandability: which method do you prefer, Extract Method or Pull Up Method?

• With respect to understandability: which method do you prefer, Extract Method or Rename Method?

• With respect to understandability: which method do you prefer, Extract Class or Inline Class?

• With respect to understandability: which method do you prefer, Extract Class or Pull Up Method?

• With respect to understandability: which method do you prefer, Extract Class or Rename Method?
• With respect to understandability: which method do you prefer, Inline Class or Pull Up Method?

• With respect to understandability: which method do you prefer, Inline Class or Rename Method?

• With respect to understandability: which method do you prefer, Pull Up Method or Rename Method?

• With respect to maintainability: which method do you prefer, Extract Method or Extract Class?

• With respect to maintainability: which method do you prefer, Extract Method or Inline Class?

• With respect to maintainability: which method do you prefer, Extract Method or Pull Up Method?

• With respect to maintainability: which method do you prefer, Extract Method or Rename Method?

• With respect to maintainability: which method do you prefer, Extract Class or Inline Class?

• With respect to maintainability: which method do you prefer, Extract Class or Pull Up Method?

• With respect to maintainability: which method do you prefer, Extract Class or Rename Method?

• With respect to maintainability: which method do you prefer, Inline Class or Pull Up Method?
• With respect to maintainability: which method do you prefer, Inline Class or Rename Method?

• With respect to maintainability: which method do you prefer, Pull Up Method or Rename Method?

• With respect to reusability: which method do you prefer, Extract Method or Extract Class?

• With respect to reusability: which method do you prefer, Extract Method or Inline Class?

• With respect to reusability: which method do you prefer, Extract Method or Pull Up Method?

• With respect to reusability: which method do you prefer, Extract Method or Rename Method?

• With respect to reusability: which method do you prefer, Extract Class or Inline Class?

• With respect to reusability: which method do you prefer, Extract Class or Pull Up Method?

• With respect to reusability: which method do you prefer, Extract Class or Rename Method?

• With respect to reusability: which method do you prefer, Inline Class or Pull Up Method?

• With respect to reusability: which method do you prefer, Inline Class or Rename Method?
With respect to reusability: which method do you prefer, Pull Up Method or Rename Method?

**Refactoring Techniques Results Based on ANP Evaluation**

Team 1’s results for ranking the refactoring patterns with respect to all four criteria is as follows: first, Extract Class; second, Extract Method; third, Rename Method; fourth, Inline Class; and fifth, Pull Up Method. Table 5.31 shows the scores for each pattern.

Team 1 ranked maintainability as the most important criterion, followed by understandability in the second position, while flexibility and reusability were ranked in the third and fourth positions, respectively. Figure 5.14 exhibits the importance of each criterion as a percentage according to Team 1.

**Table 5.31:** Ranking the refactoring patterns based on external attributes by Team 1

<table>
<thead>
<tr>
<th>Refactoring Patterns</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Class</td>
<td>32.87 %</td>
</tr>
<tr>
<td>Extract Method</td>
<td>29.51 %</td>
</tr>
<tr>
<td>Rename Method</td>
<td>14.75 %</td>
</tr>
<tr>
<td>Inline Class</td>
<td>13.62 %</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>9.24 %</td>
</tr>
</tbody>
</table>

Team 2 ranked the refactoring patterns as follows: first, Extract Class; second, Rename Method; third, Pull Up Method; fourth, Extract Method; and fifth, Inline Class. Table 5.32 displays the refactoring methods ranking according to Team 2. Furthermore, in terms of the most important criterion, Team 2 ranked understandability in the first position, while flexibility was considered a less important criterion. Table 5.33 shows the ranking of the criteria by Team 2.
Figure 5.14: The importance of the external attributes for the refactoring patterns by Team 1

Table 5.32: Refactoring methods ranking by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Refactoring Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Extract Class</td>
</tr>
<tr>
<td>2</td>
<td>Rename Method</td>
</tr>
<tr>
<td>3</td>
<td>Pull Up Method</td>
</tr>
<tr>
<td>4</td>
<td>Extract Method</td>
</tr>
<tr>
<td>5</td>
<td>Inline Class</td>
</tr>
</tbody>
</table>

Observations

1. With respect to all of the criteria, both teams ranked Extract Class in the first position.

2. Team 1 ranked Extract Method in the second position, followed by Rename Method in the third position.

3. Team 2 ranked Rename Method in the second position, followed by Pull Up Method.

4. Pull Up Method was ranked in the last position by Team 1.

5. Maintainability was considered the most important quality attribute by Team 1, while understandability was considered the most important quality attribute
Table 5.33: The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Understandability</td>
</tr>
<tr>
<td>2</td>
<td>Maintainability</td>
</tr>
<tr>
<td>3</td>
<td>Reusability</td>
</tr>
<tr>
<td>4</td>
<td>Flexibility</td>
</tr>
</tbody>
</table>

by Team 2.

6. By considering each criterion individually, we can see that Team 1 ranked Extract Class in the top position with respect to understandability. Table 5.34 shows the ranking of all refactoring techniques with respect to each external attribute.

7. In terms of the top refactoring technique with respect to understandability, Team 2 ranked the Rename Method in the top position.

8. With respect to maintainability, Team 1 ranked Extract Class in the first position, while Team 2 ranked the Pull Up Method in the top position.

9. With respect to each refactoring technique individually, we can see that understandability was ranked in the top position according to the Rename Method. Table 5.35 shows the weight of each criterion with respect to each refactoring pattern.

Refactoring Experience By Participants

The students were asked to practice the mentioned refactoring techniques during the development cycle of the XP project. In addition, we asked the students to observe the refactoring’s effect on the quality attributes. Table 5.36 illustrates some examples of the refactoring techniques used.
Table 5.34: Refactoring techniques with respect to each criterion for Team 1

<table>
<thead>
<tr>
<th>Techniques</th>
<th>maintain-ability</th>
<th>Techniques</th>
<th>Understand-ability</th>
<th>Techniques</th>
<th>flex-ibility</th>
<th>Techniques</th>
<th>reus-ability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Class</td>
<td>27.35 %</td>
<td>Extract Class</td>
<td>41.98 %</td>
<td>Extract Method</td>
<td>45.85 %</td>
<td>Extract Method</td>
<td>47.10 %</td>
</tr>
<tr>
<td>Extract Method</td>
<td>22.45 %</td>
<td>Extract Method</td>
<td>31.91 %</td>
<td>Extract Class</td>
<td>28.76 %</td>
<td>Extract Class</td>
<td>28.00 %</td>
</tr>
<tr>
<td>Inline Class</td>
<td>22.15 %</td>
<td>Rename Method</td>
<td>16.05 %</td>
<td>Pull Up Method</td>
<td>14.00 %</td>
<td>Pull Up Method</td>
<td>13.67 %</td>
</tr>
<tr>
<td>Rename Method</td>
<td>16.06 %</td>
<td>Inline Class</td>
<td>6.30 %</td>
<td>Rename Method</td>
<td>7.29 %</td>
<td>Rename Method</td>
<td>7.38 %</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>11.97 %</td>
<td>Pull Up Method</td>
<td>3.74 %</td>
<td>Inline Class</td>
<td>4.07 %</td>
<td>Inline Class</td>
<td>3.84 %</td>
</tr>
</tbody>
</table>

Table 5.35: Criteria weights with respect to each refactoring technique for Team 1

<table>
<thead>
<tr>
<th></th>
<th>Extract Method</th>
<th>Pull Up Method</th>
<th>Inline Class</th>
<th>Extract Class</th>
<th>Rename Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understandability</td>
<td>24.82 %</td>
<td>55.67 %</td>
<td>47.54 %</td>
<td>27.82 %</td>
<td>64.50 %</td>
</tr>
<tr>
<td>Maintainability</td>
<td>62.65 %</td>
<td>24.89 %</td>
<td>27.45 %</td>
<td>59.65 %</td>
<td>21.89 %</td>
</tr>
<tr>
<td>Flexibility</td>
<td>8.84 %</td>
<td>11.13 %</td>
<td>9.15 %</td>
<td>7.84 %</td>
<td>3.57 %</td>
</tr>
<tr>
<td>Reusability</td>
<td>3.66 %</td>
<td>8.29 %</td>
<td>15.84 %</td>
<td>4.66 %</td>
<td>10.02 %</td>
</tr>
</tbody>
</table>

Moreover, table 5.37 and table 5.38 show the number of refactoring patterns that have been applied by both teams in each iteration.

Observations of the Internal Effect of Refactoring

This section shows the effect of the refactoring patterns on the internal quality metrics. The students were asked to use (-) to indicate a decrease, (+) to indicate an increase, and (0) to indicate no use or no change. Table 5.39 and table 5.39 display the effect of refactoring on the internal quality attributes as reported by both teams.

Observations of the External Effect of Refactoring

Similar to the internal impacts, this section shows the impact of refactoring on the external quality attributes as reported by the two teams. Tables 5.41 and 5.42 show the effect of refactoring patterns on the external quality attributes.
Table 5.36: Example of refactoring patterns

<table>
<thead>
<tr>
<th>Refactoring Patterns</th>
<th>Where/Implementing</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Class</td>
<td>User class created and used in all C# forms</td>
<td>Users have multiple attributes. Extract class helps in managing all the attributes using single variable and it makes easy for passing variables between C# forms.</td>
</tr>
<tr>
<td>Inline Class</td>
<td>All appointment related C# forms.</td>
<td>Inline class technique is used for appointment class. It has fewer members and was not required to be passed between forms. Inline class makes it simple.</td>
</tr>
<tr>
<td>Rename Method</td>
<td>All methods are named using 'Rename Method’ Scheme</td>
<td>Easy management and debugging.</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>All buttonClick-Methods, where database is accessed.</td>
<td>All these events are different, it makes is simpler to implement and debug the code.</td>
</tr>
</tbody>
</table>

Table 5.37: Number of refactoring patterns was applied by Team 1 in each iteration

<table>
<thead>
<tr>
<th></th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Iteration 4</th>
<th>Iteration 5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Class</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>22</td>
</tr>
<tr>
<td>Inline Class</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>14</td>
</tr>
<tr>
<td>Rename Method</td>
<td>0</td>
<td>42</td>
<td>38</td>
<td>24</td>
<td>7</td>
<td>101</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Extract Method</td>
<td>0</td>
<td>19</td>
<td>13</td>
<td>11</td>
<td>16</td>
<td>59</td>
</tr>
</tbody>
</table>

Table 5.38: Number of refactoring patterns was applied by Team 2 in each iteration

<table>
<thead>
<tr>
<th></th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Iteration 4</th>
<th>Iteration 5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Class</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>Inline Class</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Rename Method</td>
<td>0</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Extract Method</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
</tbody>
</table>
### Table 5.39: Refactoring patterns effect on the internal attributes by Team 1

<table>
<thead>
<tr>
<th></th>
<th>Cohesion</th>
<th>Coupling</th>
<th>Code-Size</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Method</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Inline Class</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Extract Class</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Rename Method</td>
<td>+</td>
<td>0</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

### Table 5.40: Refactoring patterns effect on the internal attributes by Team 2

<table>
<thead>
<tr>
<th></th>
<th>Cohesion</th>
<th>Coupling</th>
<th>Code-Size</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Method</td>
<td>+</td>
<td>0</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Inline Class</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Extract Class</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Rename Method</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

### Table 5.41: Refactoring patterns effect on the external attributes by Team 1

<table>
<thead>
<tr>
<th></th>
<th>Maintainability</th>
<th>Understandability</th>
<th>Flexibility</th>
<th>Reusability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Method</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Inline Class</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Extract Class</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Rename Method</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 5.42: Refactoring patterns effect on the external attributes by Team 2

<table>
<thead>
<tr>
<th></th>
<th>Maintainability</th>
<th>Understandability</th>
<th>Flexibility</th>
<th>Reusability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Method</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Inline Class</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Extract Class</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Pull Up Method</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>Rename Method</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
5.5 ANP in Test-Driven Development

In this chapter, we introduce two uses of the ANP in test-driven development (TDD) practice. The first section of this chapter offers a brief introduction to the test driven development practice and the recent related work in the field. This chapter also shows the network structure in testing practice that includes several significant criteria and different testing techniques, as well as various release indicators. The results of applying the ANP in testing practice are presented and discussed in this chapter.

5.5.1 Introduction

In TDD, unit test cases are cumulatively written before starting implementation. The TDD practice was first explained by Beck [159]. The concept of TDD is well recommended in software life cycle as it offers several advantages, such as building high quality designs, enhancing programming productivity, minimizing the work needed to fix defects, and increasing programming speed [160].

The unit tests are written before implementation, and they have to pass every time. In XP projects, most of the tests are automated, which saves time and effort, and minimizes the overall testing cost. It is recommended to execute regressions testing in order to ensure the program is still working with the new changes. Since XP has two types of testing, unit case and acceptance testing, the testing method might be used for both of them [161].

While automated testing is certainly the core of TDD, this type of testing could be more costly than manual testing [162]. Therefore, it is significant for the XP team to know when to do automated testing and when to perform semi-automated or manual tests.

JoEllen [163] notes that the test should not be automated if it is only executed one time, as “one-time tasks and exploratory testing/edge cases should not be au-
tomated. Edge cases are, by definition, usually one-off test cases, and the effort to automate edge cases generally does not pay off. Exploratory tests are best used to gain knowledge of a new feature and then tweak or revise tests based on the new knowledge” [163].

Johnson [164] investigated the significance of skills in automated testing and discussed the idea of replacing testers through automation. The author also discussed the idea of role and skills changes that are required in order to achieve the quality verification task during automation testing. Johnson [164] stated that automation testing does not replace the need for testers; however, the author noted that automation testing changes the skills needed to fulfill the quality verification task [164].

In manual testing, testers have more freedom to examine and try to break the system, but this type of testing consumes more time. In contrast, in automation testing, it is impossible to break the system, but it notifies when the code change broke the test. Therefore, performing automation testing could be a questionable decision. Several automated attempts could not meet the expected return on investment or could fail [164].

Crispin and House [165] emphasized that when performing manual testing, the testing quality is reduced due to schedule pressure. According to Crispin and House [165] “people begin to cut corners, omit tests, and miss problems. This is the kind of dysfunctional behavior for which traditional software development is famous. All those manual test cases look great on paper, but when crunch time hits, all the paper goes out the window” [165]. The authors also stated that all acceptance tests have to be automated [165].

5.5.2 Applying ANP in Test-Driven Development

Based on the previous section, it is obvious that many criteria affect the decision to engage in fully automated testing. These criteria include developers understanding,
testing coverage, graphic user interface (GUI) applications, test execution cost, and time. The ANP, therefore, can be used to assist in selecting one of three testing methods: manual testing, semi-automated testing, and automated testing.

The second application of ANP in testing practice is ranking the release indicators in order to help the development team decide the suitable time to release the software. Gregory [164] specified several factors and non-analytical methods that can help to determine when the software is ready to be released [164]. The ANP can be applied to evaluate the release indicators in order to enhance test efficiency. The selected indicators are: percentage of defects reported, percentage of code executed with no error, and number of completed test cases. The selected indicators are evaluated with respect to simplicity, design, development cost, code quality, and team velocity.

5.5.3 First: Automated Testing Decision

The first application of the ANP in testing practice is to determine the testing automation level with respect to the proposed criteria. These criteria are discussed in the following section.

Proposed Criteria for the Automation Testing Decision

The participants were asked to evaluate the testing options with respect to the following criteria:

- **Developer Understanding**: What is the preferable testing option in terms of being easy to understand and apply by the developers?

- **Test Coverage**: What is the preferable testing option that can cover most of the tested source code?

- **Graphic User Interface (GUI)**: What is the preferable testing option in terms of dealing with GUI testing?
• **Time**: What is the preferable testing option that involves less time being consumed during the testing stage?

• **Cost of Execution**: What is the testing option that costs less when executed?

**ANP Structure for the Automation Testing Decision**

The ANP structure for testing decisions consists of three clusters. The first cluster is the objective, which determines the level of testing automation. The second cluster includes the selected criteria, which are developer understanding, test coverage, GUI, time, and cost of execution. The third cluster contains the following options: manual testing, automated testing, and semi-automated testing. Figure 5.15 shows the ANP network for the problem.

![ANP network for selecting the level of automated testing](image)

**Figure 5.15**: ANP network for selecting the level of automated testing

**Pairwise Comparisons for Deciding the Automated Level**

The ANP team was asked to evaluate the importance of each criterion with respect to all of the testing options in order to show the interdependencies. In addition, the
team was asked to evaluate each testing option with respect to all of the criteria in order to show the feedback of the network. The Team 1 participants received the suitable ANP tables in order to make it easy for them to perform the pairwise comparisons based on the ANP fundamental scale that was described previously. For the interdependencies relationship, the participants were asked the following:

- With respect to Automated Testing: which criterion is more important, developer understanding or test coverage and by how much?

- With respect to Automated Testing: which criterion is more important, developer understanding or GUI and by how much?

- With respect to Automated Testing: which criterion is more important, test coverage or time and by how much?

- With respect to Automated Testing: which criterion is more important, test coverage or GUI and by how much?

- With respect to Automated Testing: which criterion is more important, cost of execution or developer understanding and by how much?

- With respect to Automated Testing: which criterion is more important, cost of execution or GUI and by how much?

- With respect to Automated Testing: which criterion is more important, time or cost of execution and by how much?

- With respect to Automated Testing: which criterion is more important, time or GUI and by how much?

- With respect to Automated Testing: which criterion is more important, time or developer understanding and by how much?
- With respect to Automated Testing: which criterion is more important, test coverage or cost of execution and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, developer understanding or test coverage and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, developer understanding or GUI and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, test coverage or time and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, test coverage or GUI and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, cost of execution or developer understanding and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, cost of execution or GUI and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, time or cost of execution and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, time or GUI and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, time or developer understanding and by how much?

- With respect to Semi-Automated Testing: which criterion is more important, test coverage or cost of execution and by how much?
• With respect to Manual Testing: which criterion is more important, developer understanding or test coverage and by how much?

• With respect to Manual Testing: which criterion is more important, developer understanding or GUI and by how much?

• With respect to Manual Testing: which criterion is more important, test coverage or time and by how much?

• With respect to Manual Testing: which criterion is more important, test coverage or GUI and by how much?

• With respect to Manual Testing: which criterion is more important, cost of execution or developer understanding and by how much?

• With respect to Manual Testing: which criterion is more important, cost of execution or GUI and by how much?

• With respect to Manual Testing: which criterion is more important, time or cost of execution and by how much?

• With respect to Manual Testing: which criterion is more important, time or GUI and by how much?

• With respect to Manual Testing: which criterion is more important, time or developer understanding and by how much?

• With respect to Manual Testing: which criterion is more important, test coverage or cost of execution and by how much?

The participants then compared the three testing options with respect to each criterion in order to evaluate the feedback relationship. Examples of these questions for the participants are:
• With respect to developer understanding: which testing option do you prefer, automated testing or semi-automated testing?

• With respect to developer understanding: which testing option do you prefer, automated testing or manual testing?

• With respect to developer understanding: which testing option do you prefer, semi-automated testing or manual testing?

• With respect to test coverage: which testing option do you prefer, automated testing or semi-automated testing?

• With respect to test coverage: which testing option do you prefer, automated testing or manual testing?

• With respect to test coverage: which testing option do you prefer, semi-automated testing or manual testing?

• With respect to time: which testing option do you prefer, automated testing or semi-automated testing?

• With respect to time: which testing option do you prefer, automated testing or manual testing?

• With respect to time: which testing option do you prefer, semi-automated testing or manual testing?

• With respect to GUI: which testing option do you prefer, automated testing or semi-automated testing?

• With respect to GUI: which testing option do you prefer, automated testing or manual testing?
• With respect to GUI: which testing option do you prefer, semi-automated testing or manual testing?

• With respect to cost of execution: which testing option do you prefer, automated testing or semi-automated testing?

• With respect to cost of execution: which testing option do you prefer, automated testing or manual testing?

• With respect to cost of execution: which testing option do you prefer, semi-automated testing or manual testing?

Automation Decision Results Based on ANP Evaluation

Team 1’s results show that the automated option was given the highest rank among the three alternatives. The manual testing option came second, followed by the semi-automated option. Table 5.43 shows the relative weight of each testing option as a percentage. In addition, in using the software, we were able to examine the importance of each criterion based on all of the alternatives. Test coverage was ranked as being of the highest importance among the five criteria, followed by time, developer understanding, GUI, and cost of execution. Figure 5.16 shows the criteria importance scores.

Table 5.43: Automated testing level for Team 1

<table>
<thead>
<tr>
<th>Technique</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated</td>
<td>58.58 %</td>
</tr>
<tr>
<td>Manual</td>
<td>23.71 %</td>
</tr>
<tr>
<td>Semi-Automated</td>
<td>17.69 %</td>
</tr>
</tbody>
</table>

Team 2’s results show that the team ranked the automated testing option as the highest alternative, followed by semi-automated and manual testing. Table 5.44 shows
the testing options ranking by Team 2. In terms of the most important factor for
deciding the level of automation, Team 2 gave cost of execution the top score. Table
5.45 shows the ranking of the criteria by Team 2.

Table 5.44: Automated testing level for Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Testing Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Automation</td>
</tr>
<tr>
<td>2</td>
<td>Semi-Automation</td>
</tr>
<tr>
<td>3</td>
<td>Manual</td>
</tr>
</tbody>
</table>

Table 5.45: The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cost of Execution</td>
</tr>
<tr>
<td>2</td>
<td>Time</td>
</tr>
<tr>
<td>3</td>
<td>Code Coverage</td>
</tr>
<tr>
<td>4</td>
<td>Developer Understanding</td>
</tr>
<tr>
<td>5</td>
<td>GUI</td>
</tr>
</tbody>
</table>

Observations

• With respect to all of the criteria, Team 1 chose automated testing as the best
  testing option.
• In terms of the best testing option, Team 2 selected the automated option.

• Team 1 ranked manual testing as the second option, followed by semi-automated testing option as the last option.

• Team 2 ranked semi-automated testing as the second option, followed by manual testing as the last option.

• Test coverage was considered the most important criteria by Team 1, while cost of execution was selected as the most important criteria by Team 2.

• Considering each criterion individually, we can note that Team 1 considered automated testing as the top option with respect to time, test coverage, and cost of execution criteria, while manual testing was considered as the best option with respect to GUI and developer understanding criteria. Table 5.46 shows the testing options ranking with respect to each criterion.

• In terms of the best testing option with respect to developer understanding criterion, Team 2 selected manual testing as the top option.

• With respect to each testing technique individually, we can see that test coverage criterion was ranked in the top position according to the semi-automated testing option. Table 5.47 shows the weight of each criterion with respect to each refactoring pattern.
Table 5.46: Automated testing level with respect to each criterion individually for Team 1

<table>
<thead>
<tr>
<th>Technique</th>
<th>Test Coverage</th>
<th>Technique</th>
<th>Time</th>
<th>Technique</th>
<th>GUI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automation</td>
<td>78.70 %</td>
<td>Automation</td>
<td>72.91 %</td>
<td>Manual</td>
<td>77.21 %</td>
</tr>
<tr>
<td>Semi-Automation</td>
<td>16.72 %</td>
<td>Semi-Automation</td>
<td>19.95 %</td>
<td>Semi-Automation</td>
<td>17.31 %</td>
</tr>
<tr>
<td>Manual</td>
<td>4.57 %</td>
<td>Manual</td>
<td>10.34 %</td>
<td>Automation</td>
<td>5.47 %</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technique</th>
<th>Developer Understanding</th>
<th>Technique</th>
<th>Cost of Execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td>71.47 %</td>
<td>Automation</td>
<td>68.70 %</td>
</tr>
<tr>
<td>Semi-Automation</td>
<td>21.84 %</td>
<td>Semi-Automation</td>
<td>19.72 %</td>
</tr>
<tr>
<td>Automation</td>
<td>6.68 %</td>
<td>Manual</td>
<td>9.57 %</td>
</tr>
</tbody>
</table>

Table 5.47: Criteria weights with respect to each testing option for Team 1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Coverage</td>
<td>33.68 %</td>
<td>49.63 %</td>
<td>36.03 %</td>
</tr>
<tr>
<td>Time</td>
<td>44.12 %</td>
<td>11.89 %</td>
<td>3.90 %</td>
</tr>
<tr>
<td>GUI</td>
<td>2.75 %</td>
<td>19.16 %</td>
<td>15.78 %</td>
</tr>
<tr>
<td>Developer Understanding</td>
<td>11.02 %</td>
<td>15.69 %</td>
<td>40.86 %</td>
</tr>
<tr>
<td>Cost of Execution</td>
<td>8.41 %</td>
<td>3.61 %</td>
<td>3.41 %</td>
</tr>
</tbody>
</table>
5.5.4 Second: Ranking Release Indicators

Ranking the release indicators is the second application of the ANP in the TDD practice. Three release indicators were selected in order to rank them based on their importance. These three indicators were introduced by Gregory [164] to assist the development team in evaluating the software in terms of its readiness to be released. The proposed indicators are: percentage of defects reported, percentage of code executed without error, and number of test cases completed. These indicators will be examined according to the following proposed criteria.

Proposed Criteria for Ranking Release Indicators

The participants were asked to evaluate the testing options with respect to the following criteria:

- **Velocity:** What is the preferable release indicator in terms of speeding up the team velocity?

- **Simplicity:** What is the preferable release indicator that is simpler to apply and complete?

- **Design:** Which release indicator alternative will improve the design?

- **Cost:** Which release indicator alternative costs more in terms of money and effort?

- **Code Quality:** Which release indicator alternative can better enhance the quality of code?

ANP Structure for Ranking Release Indicators

The ANP structure for ranking the release indicators consists of three clusters. The first cluster is the objective, which involves finding the most significant release indi-
cator. The second cluster includes the selected criteria, which are velocity, simplicity, design, cost, and code quality. The third cluster contains the following indicators: percentage of defects reported, percentage of code executed without error, and number of test cases completed. Figure 5.17 shows the ANP network for the problem.

![ANP network for ranking the release indicators](image)

**Figure 5.17:** ANP network for ranking the release indicators

### Pairwise Comparisons for Ranking Release Indicators

The ANP team was asked to evaluate the importance of each criterion with respect to all of the release indicators, in order to show the interdependencies. In addition, the team was asked to evaluate each release indicator alternative with respect to all of the criteria, in order to show the feedback of the network. The Team 1 participants received the suitable ANP tables in order to make it easy for the participants to perform the pairwise comparisons based on the ANP fundamental scale that was described previously. For the interdependencies relationship, the participants were asked the following:

- With respect to number of test cases completed: which criterion is more impor-
tant, velocity or design and by how much?

- With respect to number of test cases completed: which criterion is more important, velocity or cost and by how much?

- With respect to number of test cases completed: which criterion is more important, design or simplicity and by how much?

- With respect to number of test cases completed: which criterion is more important, design or cost and by how much?

- With respect to number of test cases completed: which criterion is more important, code quality or velocity and by how much?

- With respect to number of test cases completed: which criterion is more important, code quality or cost and by how much?

- With respect to number of test cases completed: which criterion is more important, simplicity or code quality and by how much?

- With respect to number of test cases completed: which criterion is more important, simplicity or cost and by how much?

- With respect to number of test cases completed: which criterion is more important, simplicity or velocity and by how much?

- With respect to number of test cases completed: which criterion is more important, design or code quality and by how much?

- With respect to percentage of defects reported: which criterion is more important, velocity or design and by how much?

- With respect to percentage of defects reported: which criterion is more important, velocity or cost and by how much?
• With respect to percentage of defects reported: which criterion is more important, design or simplicity and by how much?

• With respect to percentage of defects reported: which criterion is more important, design or cost and by how much?

• With respect to percentage of defects reported: which criterion is more important, code quality or velocity and by how much?

• With respect to percentage of defects reported: which criterion is more important, code quality or cost and by how much?

• With respect to percentage of defects reported: which criterion is more important, simplicity or code quality and by how much?

• With respect to percentage of defects reported: which criterion is more important, simplicity or cost and by how much?

• With respect to percentage of defects reported: which criterion is more important, simplicity or velocity and by how much?

• With respect to percentage of defects reported: which criterion is more important, design or code quality and by how much?

• With respect to percentage of code executed without error: which criterion is more important, velocity or design and by how much?

• With respect to percentage of code executed without error: which criterion is more important, velocity or cost and by how much?

• With respect to percentage of code executed without error: which criterion is more important, design or simplicity and by how much?
• With respect to percentage of code executed without error: which criterion is more important, design or cost and by how much?

• With respect to percentage of code executed without error: which criterion is more important, code quality or velocity and by how much?

• With respect to percentage of code executed without error: which criterion is more important, code quality or cost and by how much?

• With respect to percentage of code executed without error: which criterion is more important, simplicity or code quality and by how much?

• With respect to percentage of code executed without error: which criterion is more important, simplicity or cost and by how much?

• With respect to percentage of code executed without error: which criterion is more important, simplicity or velocity and by how much?

• With respect to percentage of code executed without error: which criterion is more important, design or code quality and by how much?

The participants then compared the three release indicators with respect to each criterion, in order to evaluate the feedback relationship. Examples of these questions for the participants are:

• With respect to simplicity: which release indicator do you prefer, number of test cases completed or percentage of defects reported?

• With respect to simplicity: which release indicator do you prefer, number of test cases completed or percentage of code executed without error?

• With respect to simplicity: which release indicator do you prefer, percentage of defects reported or percentage of code executed without error?
• With respect to design: which release indicator do you prefer, number of test cases completed or percentage of defects reported?

• With respect to design: which release indicator do you prefer, number of test cases completed or percentage of code executed without error?

• With respect to design: which release indicator do you prefer, percentage of defects reported or percentage of code executed without error?

• With respect to velocity: which release indicator do you prefer, number of test cases completed or percentage of defects reported?

• With respect to velocity: which release indicator do you prefer, number of test cases completed or percentage of code executed without error?

• With respect to velocity: which release indicator do you prefer, percentage of defects reported or percentage of code executed without error?

• With respect to cost: which release indicator do you prefer, number of test cases completed or percentage of defects reported?

• With respect to cost: which release indicator do you prefer, number of test cases completed or percentage of code executed without error?

• With respect to cost: which release indicator do you prefer, percentage of defects reported or percentage of code executed without error?

• With respect to code quality: which release indicator do you prefer, number of test cases completed or percentage of defects reported?

• With respect to code quality: which release indicator do you prefer, number of test cases completed or percentage of code executed without error?
With respect to code quality: which release indicator do you prefer, percentage of defects reported or percentage of code executed without error?

Ranking Release Indicators Results Based on ANP Evaluation

Team 1’s results show that the percentage of code executed without error indicator was given the highest rank among the three alternatives. The number of test cases completed came second, followed by percentage of defects reported. Table 5.48 shows the relative weight of each release indicator as a percentage. In addition, by using the software, we were able to examine the importance of each criterion based on all of the alternatives. The code quality was ranked as being of the highest importance among the five criteria, followed by design, simplicity, cost, and velocity. Figure 5.18 shows the criteria importance scores.

Table 5.48: Automated testing level for Team 1

<table>
<thead>
<tr>
<th>Release Indicators</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of code executed without error</td>
<td>57.39 %</td>
</tr>
<tr>
<td># of test cases completed</td>
<td>27.91 %</td>
</tr>
<tr>
<td>% of defects reported</td>
<td>14.68 %</td>
</tr>
</tbody>
</table>

Figure 5.18: The importance of the criteria for Team 1 in release indicators
Team 2’s results show that the team ranked the percentage of code executed without error indicator as the highest alternative, followed by percentage of defects reported and number of test cases completed. Table 5.49 shows the release indicators ranking by Team 2. In terms of the most important factor for ranking the release indicators, Team 2 gave the code quality criterion the top position. Table 5.50 shows the ranking of the criteria by Team 2.

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Release Indicators</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>% Of Execution Without Error</td>
</tr>
<tr>
<td>2</td>
<td>% Of Defects Reported</td>
</tr>
<tr>
<td>3</td>
<td># Of Test Cases Completed</td>
</tr>
</tbody>
</table>

Table 5.50: The importance of the criteria by Team 2

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Code Quality</td>
</tr>
<tr>
<td>2</td>
<td>Cost</td>
</tr>
<tr>
<td>3</td>
<td>Simplicity</td>
</tr>
<tr>
<td>4</td>
<td>Design</td>
</tr>
<tr>
<td>5</td>
<td>Velocity</td>
</tr>
</tbody>
</table>

Observations

1. With respect to all of the criteria, both teams ranked the percentage of code executed without error indicator in the highest position.

2. The number of test cases completed indicator was ranked in the second position by Team 1, while Team 2 ranked it in the lowest position.

3. Both teams ranked code quality as the most important criterion, and both teams also ranked velocity criterion in the lowest position.
4. Considering each criterion individually, we can note that Team 1 ranked the number of test cases completed indicator at the top with respect to the design and simplicity criteria. Table 5.51 shows the ranking of the three release indicators with respect to each criterion.

5. Regarding the top release indicator in terms of simplicity, Team 2 ranked the number of test cases completed indicator at the top position.

6. If we consider the criteria ranking based on each release indicator individually, we can see that Team 1 ranked the code quality criterion in the top position with respect to the three release indicators. Table 5.52 exhibits the criteria ranking based on each release indicator.
Table 5.51: Release indicators by Team 1 with respect to each criterion individually

<table>
<thead>
<tr>
<th>Release Indicators</th>
<th>Code Quality</th>
<th>Release Indicators</th>
<th>Cost</th>
<th>Release Indicators</th>
<th>Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of code executed without error</td>
<td>77.91 %</td>
<td>% of code executed without error</td>
<td>72.19 %</td>
<td># of test cases completed</td>
<td>73.83 %</td>
</tr>
<tr>
<td>% of defects reported</td>
<td>16.09 %</td>
<td>% of defects reported</td>
<td>22.70 %</td>
<td>% of code executed without error</td>
<td>17.01 %</td>
</tr>
<tr>
<td># of test cases completed</td>
<td>5.98 %</td>
<td># of test cases completed</td>
<td>5.10 %</td>
<td>% of defects reported</td>
<td>9.15 %</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Release Indicators</th>
<th>Simplicity</th>
<th>Release Indicators</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of code executed without error</td>
<td>75.31 %</td>
<td>% of code executed without error</td>
<td>67.98 %</td>
</tr>
<tr>
<td>% of defects reported</td>
<td>18.39 %</td>
<td>% of defects reported</td>
<td>22.02 %</td>
</tr>
<tr>
<td>% of defects reported</td>
<td>6.29 %</td>
<td># of test cases completed</td>
<td>9.99 %</td>
</tr>
</tbody>
</table>

Table 5.52: Criteria weights for Team 1 with respect to each release indicator

<table>
<thead>
<tr>
<th></th>
<th>% of code executed without error</th>
<th>% of defects reported</th>
<th># of test cases completed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code quality</td>
<td>51.04 %</td>
<td>45.40 %</td>
<td>54.23 %</td>
</tr>
<tr>
<td>Cost</td>
<td>7.07 %</td>
<td>9.78 %</td>
<td>12.53 %</td>
</tr>
<tr>
<td>Velocity</td>
<td>10.89 %</td>
<td>7.87 %</td>
<td>2.66 %</td>
</tr>
<tr>
<td>Design</td>
<td>28.15 %</td>
<td>28.14 %</td>
<td>5.18 %</td>
</tr>
<tr>
<td>Simplicity</td>
<td>2.83 %</td>
<td>8.78 %</td>
<td>25.38 %</td>
</tr>
</tbody>
</table>
5.6 Analysis and Discussion

This section presents the comments and feedback that were acquired from the participants based on the application of ANP in the XP development cycle. First, this section presents the semi-structured interview answers. Second, the questionnaires results are summarized. Third, the research questions and collected data are linked with the propositions. Finally, the criteria are applied for the interpretation of the findings.

5.6.1 Semi-Structure Interview

After completing the project, the results of the ANP evaluation of all XP practices were shown to the participants in order to conduct the interviews. Not all results were as expected and some findings were surprising. The interviews involved open-ended questions used to collect the participants’ perspectives about the ANP, its benefits and disadvantages in XP, and their opinions about the best application for ANP in XP among all mentioned practices. The collected data was comprised of handwritten notes from the interviews. These notes were arranged in a folder in order to be easily reached and analyzed. The semi-structured interview questions and answers are presented below.

1. What do you like about the ANP?

- “The analytic network process is an easy tool to use in extreme programming. I like the simplicity of using it in all extreme programming practices. Also, it is easy to understand within a short time and with less effort”.

- “The analytic network process is a powerful tool that can make complicated and unstructured problems very easy to be solved. The ANP is good because it deals with multi-criteria decision making problems”. 
• “I like the analytic network process because it organizes the criteria and alternatives in clear clusters, which makes it easy to make more accurate decisions”.

• “The pairwise comparisons are a very nice way to decide the best alternative among different alternatives”.

• “The ANP is good to save the XP team time in making their decisions during the development process. All the team members’ opinions need to be considered, so the ANP will formalize this process in a nice way”.

• “The analytic network process helps me in more understanding of the problem and, also, it introduces a good way to help me in solving the problem with the best solution”.

2. In which XP practice do you think the ANP was most beneficial? Why?

• “For me, I think in pair programming practice because the matching based on the ANP helped me a lot with my partner”.

• “The most beneficial use of the ANP was in planning game and refactoring. I prefer it in these two practices regarding the formalization of the decision in planning game and refactoring activities”.

• “In my opinion, the ANP was very beneficial in the design practice. it helped the team in selecting the most appropriate design tool, and this helped in completing the design. In this stage the ANP helped the team to communicate easily with each other, and gave us a roadmap to select the most suitable design tool. Also, I think the ANP was very beneficial in refactoring practice”.

• “The ANP was beneficial in all XP areas; however, planning game and pair programming were more beneficial that other practices”.

197
3. What are the advantages that you found by applying the ANP in XP?

- “The pairwise comparisons are very clear to make judgements”.
- “It helps me in considering several factors that affect the project during making decisions”.
- “ANP solves multi-criteria decision making problem as required by XP and, also, it considers the effect of each criterion on each other, which makes our decisions more accurate”.
- “The ANP helped the team in having positive discussions, which is good in making a good progress. Also, all the team members were able to participate in the decision process, and everyone’s voice was heard”.
- “The mathematical basis of the ANP makes it a very powerful tool to make more reliable decisions”.
- “The most great advantage of the ANP is that it can solve complex problems in an easy way”.

4. What are the drawbacks that you found by applying the ANP in XP?

- “It requires all team members to participate equally in order to achieve the best results”.
- “It’s hard to answer regarding the different application areas; however, it is time consuming”.
• “I think the ANP is good for experts from the industrial environments not with educational. It is hard to follow all these steps within limited time and other academic tasks that I want to do during the semester”.

• “I think the amount of discussion that is provided by the ANP is too much, which takes more time from the project development cycle. This is not good. We need to have a good discussion, but within limited time in order to do other project tasks”.

• “The number of pairwise comparisons is too large, I think this is the main disadvantage of this tool”.

• “The individual judgments sometimes are lost regarding the aggregation of the entire team results”.

5. Would you apply the ANP if you were a decision maker in XP development?

• “Definitely, I would apply the ANP. It is helpful in facilitating the decision process in XP development”.

• “I would use it if the communication among the team members is not going good. In this case I would use it to improve the team communication”.

• “Yes, I would use it in order to remove any uncertainty during the decision making process”.

• “Yes, I would recommend it, but with small team size”.

• “Yes, I would use it while considering the available time”.

6. Do you have any suggestions for applying the ANP in XP?

• “It is good to think about it as a mobile tool”.

• “Add more considerations for the project manager in order to balance the resource leveling during the project development”.

199
• “Apply the ANP in a real XP project in industrial environment and investigate its application within a different environment”.

• “Reducing the number of pairwise comparisons”.

The interview results show positive comments from the students regarding the ANP. The ANP was a helpful tool for resolving conflicting perspectives, and encouraged each team member to participate in making decisions. The main concerns were the time it took during the ANP evaluation and the number of pairwise comparisons. All ANP team members recommended using ANP in future XP projects and trying to investigate the adoption of the ANP in an industrial environment.

5.6.2 Questionnaires

Questionnaires were distributed among the participants in order to collect their experiences and viewpoints regarding ANP. The questionnaires consisted of two sections. The first section included questions about ANP as a ranking and decision tool, as follows:

1. Examine the ANP as a decision approach applied in XP based on two features, which are practicality and quality of a decision.

The second section included questions about the benefits of each XP practice, and the students’ satisfaction.

1. Examine the ANP’s influence on user satisfaction and team for each XP practice.

In this study, a seven-point Likert scale was used to determine the acceptability level of the ANP tool, as follows:

1. Totally unacceptable.

2. Unacceptable.
3. Slightly unacceptable.


5. Slightly acceptable.

6. Acceptable.


After participants completed the questionnaire, the results were calculated. Table 5.55 shows the total acceptability percentage for each statement.

The total acceptability percentage can be obtained as follows:

The total acceptability percentage (TAP) = the average of the score for the team \( \times \frac{100}{7} \).

Where the average of the score for the team = the sum of all scores given by team members/number of team members.

The tables below exhibit the level of acceptability for the ANP following a ranking and decision approach. Furthermore, the tables show the level of acceptability of the ANP in each XP practice. These results were obtained from the educational case study that was conducted to investigate the ANP’s ability to be applied in XP practices. Tables are curved down to the closest integer.

1. **ANP level of acceptability as a decision approach**

The tables 5.53 and 5.54 show the results of the ANP acceptability based on decision quality and practical aspects, respectively. Both tables show positive results, but time efficiency and reliability were given the lowest percentages, at (63%) and (68%), respectively.

1. **ANP level of acceptability as a ranking approach**
Table 5.53: ANP level of acceptability as a decision approach for decision quality aspect by Team 1

<table>
<thead>
<tr>
<th>Decision Quality</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clearness of the decision process</td>
<td>91%</td>
</tr>
<tr>
<td>Goodness of the decision framework</td>
<td>79%</td>
</tr>
<tr>
<td>Catching the required information</td>
<td>82%</td>
</tr>
<tr>
<td>Clearness of criteria included</td>
<td>91%</td>
</tr>
<tr>
<td>Clearness of alternatives included</td>
<td>86%</td>
</tr>
</tbody>
</table>

Table 5.54: ANP level of acceptability as a decision approach for practically aspect by Team 1

<table>
<thead>
<tr>
<th>Practically</th>
<th>Scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplicity</td>
<td>87%</td>
</tr>
<tr>
<td>Reliability</td>
<td>68%</td>
</tr>
<tr>
<td>Understandability</td>
<td>85%</td>
</tr>
<tr>
<td>Time Efficiency</td>
<td>63%</td>
</tr>
</tbody>
</table>

Table 5.55 shows the overall effects of the ANP on Team 1. In general, the impacts were positive, and the team gave the lowest percentage of (61%) regarding their satisfaction with the impact of the ANP on the release indicators.

5.6.3 Logic Linking of the Data to the Research Questions and Propositions

This section presents the logic linking the assembled data to the study propositions and research questions. Table 5.56 shows the connections between the data and the case study propositions.

5.6.4 Interpretation of the results

The data was collected from various data sources. By comparing the collected data with the study propositions, based on the interpretation of the criteria that was men-
tioned in section 4.2, we will analyze this collected data. The followings are the study propositions and their answers:

**Proposition 1:** The ANP catches significant criteria and alternatives that have an effect in each proposed XP practice. Also, the results of using the ANP display the order of alternatives based on their importance.

- The structure of the ANP can be applied in the following areas:
  1. Ranking the prioritization methods.
  2. Ranking the estimation methods.
  3. Selecting the design tool.
  4. Selecting the best pairs.
  5. Ranking the pair matching rules.
  6. Ranking the refactoring patterns according to the internal quality attributes.
  7. Ranking the refactoring patterns according to the external quality attributes.
  8. Deciding the automated testing level.
  9. Ranking the release indicators.

From the previous areas, we can see that both the alternatives and criteria are structured sufficiently, and considered. These ANP structures are shown in figures 5.1, 5.3, 5.5, 5.7, 5.9, 5.11, 5.13, 5.15, and 5.17.

The results and objectives of each ANP application in each XP practice can be summarized as follows:

- Table 5.1 exhibited the ranking of the ANP team in order to rank the XP prioritization techniques, and Kano model was ranked the highest.
The estimation techniques were ranked in table 5.6, and the expert opinion method was ranked the highest by Team 1.

In simple design practice, table 5.11 showed that the whiteboard tool was chosen as the best design tool.

For pair programming, table 5.16 displayed that the expert-expert pairing option was ranked the highest among all pairing options.

Table 5.21 showed the results for the rules of matching two programmers, and almost identical was selected as the highest alternative.

Ranking the refactoring techniques based on the internal attributes was shown in table 5.26, and the results showed that the extract method was ranked the highest by the ANP team.

Table 5.31 illustrated that the extract class refactoring technique was ranked the highest with respect to the external attributes.

In testing practice, table 5.43 exhibited that the automated level of testing was ranked the highest by the ANP team.

Finally, the release indicators ranking was shown in table 5.48, and the execution without error indicator was ranked at the top.

**Proposition 2:** The ANP helps with various decision-making problems, such as prioritization, selection, and ranking when it is required in XP practices.

- The ANP received positive feedback regarding various aspects by the ANP team. This can be seen in table 5.53 and table 5.54. These two tables displayed the ANP level of acceptability as a decision-making approach. The feedback was positive in terms of catching the required information 82%, clearness of criteria included 91%, clearness of alternatives included 86%, quality of the decision framework 79%, clearness of the decision process 91%, simplicity 87%, reliability 68%, understandability 85%, and time efficiency 63%.
Proposition 3: *The ANP includes creative debate and enhances team communication.*

- Table 5.55 supported this proposition based on the following ANP uses: prioritization methods 71%, estimation methods 72%, design tools 85%, best pair 91%, matching rules 84%, refactoring 79%, automated testing 75%, and release indicators 72%.

- Section 5.6.1 presented the semi-structured interviews that also provided positive feedback in response to questions 1, 2 and 3. The answers to these three questions support this study proposition.

Proposition 4: *The ANP clears up conflicting perspectives within the development team within a certain practice.*

- The ANP team supported this in table 5.55 based on the following ANP uses: prioritization methods 89%, estimation methods 87%, design tools 77%, best pair 81%, matching rules 77%, refactoring 70%, automated testing 72%, and release indicators 77%.

- Section 5.6.1 presented the semi-structured interviews that also provided positive feedback to questions 1, 2 and 3. The answers to these three questions support this study proposition.
### Table 5.55: ANP level of acceptability as a ranking approach for Team 1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Enhancing team communication</td>
<td>75 %</td>
<td>82 %</td>
<td>86 %</td>
<td>90 %</td>
<td>89 %</td>
<td>80 %</td>
<td>77 %</td>
</tr>
<tr>
<td>Supporting positive discussion and learning chances</td>
<td>72 %</td>
<td>72 %</td>
<td>85 %</td>
<td>91 %</td>
<td>84 %</td>
<td>79 %</td>
<td>75 %</td>
</tr>
<tr>
<td>Defining the ranking problem</td>
<td>93 %</td>
<td>91 %</td>
<td>86 %</td>
<td>80 %</td>
<td>87 %</td>
<td>81 %</td>
<td>80 %</td>
</tr>
<tr>
<td>Clearing up conflict perspectives among the developers</td>
<td>89 %</td>
<td>87 %</td>
<td>77 %</td>
<td>81 %</td>
<td>77 %</td>
<td>70 %</td>
<td>72 %</td>
</tr>
<tr>
<td>Maximizing team performance</td>
<td>77 %</td>
<td>87 %</td>
<td>75 %</td>
<td>83 %</td>
<td>79 %</td>
<td>80 %</td>
<td>71 %</td>
</tr>
<tr>
<td>Satisfaction of the ANP final results</td>
<td>71 %</td>
<td>80 %</td>
<td>78 %</td>
<td>81 %</td>
<td>73 %</td>
<td>73 %</td>
<td>74 %</td>
</tr>
<tr>
<td>Satisfaction of the effects on the XP practice</td>
<td>73 %</td>
<td>82 %</td>
<td>86 %</td>
<td>80 %</td>
<td>76 %</td>
<td>70 %</td>
<td>82 %</td>
</tr>
</tbody>
</table>

### Table 5.56: Logic linking of the data to the propositions and research questions

<table>
<thead>
<tr>
<th>Propositions</th>
<th>Collected data from the XP project</th>
<th>Questions in the semi-interview</th>
<th>Questions in the questionnaire</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sections 5.1</td>
<td>1, 2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>5.2 5.3 5.4, and 5.5</td>
<td>1, 2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>2, 3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>2, 3</td>
<td>2</td>
</tr>
</tbody>
</table>
5.7 Validity

In this section, related threats to the validity are explained. These threats can affect construct validity, external validity, internal validity, and reliability. Several researchers have emphasized that case studies are difficult to analyze due to biases and validity threats, as described in [166]: “empirical studies in general and case studies in particular are prone to biases and validity threats that make it difficult to control the quality of the study to generalize its results” [166].

5.7.1 Construct Validity

Construct validity ensures that “the treatment reflects the construct of the cause well, and the outcome reflects the construct of the effect well” [167]. It deals with matching the concept being researched and studied to the specific measurements. The small number of participants is the main threat to this case study.

Using various methods to ensure the validity of the results reduced this threat. Some of these methods are:

- Data triangulation: a major advantage of case studies is the opportunity to use several sources of evidence [168]. An evidence chain is built using interviews and surveys with various types of participants who have different skills and experience levels, and the use of participants’ comments and many observations. Therefore, a valid conclusion can be reached.

- Methodological triangulation: engaging a combination of research methods, such as conducting an XP project to serve the study purpose, surveys, results of ANP pairwise comparisons, researchers’ observations, and interviews.

- Member checking: showing the findings to the participants is recommended. This was addressed by presenting the final findings to all students in order to guarantee the accuracy of the study and to avoid researcher bias.
5.7.2 Internal Validity

Internal validity is about making sure the outcome is caused by the treatment (the effect). This type of validity is only related to explanatory case studies. This issue may be addressed by linking all data sources regarding the research questions, and linking the research questions to research propositions.

5.7.3 External Validity

External validity ensures the relationship between the construct and the effect, in order to guarantee that the experiment will be generalized to a different scope \[167\]. In this study, an additional case study will need to be conducted in different environments, such as the industrial environment, in order to involve more experts from the field. Conducting such a case study will help by facilitating a comparison of the various results and findings from different environments. Future work will increase external validity.

5.7.4 Reliability

Reliability deals with the procedure of data collection and findings. Other researchers, when following the same procedures, should arrive at similar conclusions and results. This can be achieved through the availability of the same research questions, data collection, and case studies designed by other researchers.
Chapter 6

Conclusion and Future Work

This chapter presents conclusions and suggestions for future research. Section 6.1 summarizes the ANP and its applications in the XP life cycle. Discussions of possible future work are presented in section 6.2.

6.1 Summary

This research introduces a framework to the XP team based on the ANP. The ANP can be used in several XP practices in order to make decisions and prioritize problems. This research shows that nine areas that are related to the development cycle can benefit from the ANP. This section repeats the areas of contributions with additional justification.

The ANP is a powerful multi-criteria decision-making tool that can be used to solve complex problems. The network structure allows decision makers to represent the problem more precisely. Also, the connections between the network elements help the decision makers emphasize all possible influences between the elements. In XP, team members encounter various issues that require decisions to be made, such as evaluating the importance of the components that have an effect on the XP life cycle and productivity. The ANP could be used to analyze complex decisions and
increase XP achievements. Also, the ANP helps the development team to create a collaborative decision process environment by allowing each member to have his or her own decision perspective.

This research introduces the ANP to several XP practices, as mentioned in chapter 5. For example, the ANP is used in planning game practice in order to rank prioritization techniques and estimation techniques. The ANP allows the team members to specify the factors that affect the selection of the most suitable prioritization technique. Also, the team was able to select the most preferable prioritization method while considering various criteria, such as accuracy, simplicity, collaboration, and time, which offers several benefits to both developers and stakeholders. In our case study, the ANP team ranked the Kano model as the most preferable prioritization technique, while the traditional team ranked MoScoW as the top prioritization technique. The traditional team selected this technique after deep discussions and voting while considering the time factor only, while the ANP team selected their technique based on a scientific approach while considering various factors. In the end, both teams were satisfied with their decisions and the results. The second application was to rank the XP estimation techniques in order to estimate each user story. Similar to the prioritization use, the ANP team ranked the estimation techniques based on several factors, such as time, simplicity, accuracy, and collaboration. The traditional team ranked the analogy technique as the most suitable estimation technique, while Team 1 ranked expert opinion at the top. The ANP team was able to evaluate each estimation technique from different perspectives, and to mathematically reconcile the different members’ opinions.

In XP simple design practice, the ANP has been used to select the most suitable design tool. The simple design tools were compared with each other based on four criteria, which are communication, portability, simplicity, and documentation. Team 1 selected the whiteboard as their simple design tool, while Team 2 chose the UML
tool. Both teams found their selection helpful during the design phase.

In pair programming, the ANP is applied to choose the best matching pairs and to find out the best rules for matching pairs in terms of having same or different characteristics. In selecting the best pairs, both teams found that the expert-expert pair is the best matching with respect to the four criteria, which are sharing knowledge, speed, code quality, and learning. In the second use of ANP in pair programming practice, the participants evaluated the three alternatives mentioned in chapter 5.3 with respect to personality, gender, experience, programming style, and culture.

Regarding refactoring, the ANP was used for two purposes. First, the ANP was used in order to rank refactoring patterns with respect to the internal code quality attributes. The selected internal code quality attributes were: cohesion, coupling, code size, and complexity. Both teams ranked the Extract Class and Extract Method in the top positions. Second, the ANP was used in order to rank refactoring patterns with respect to the external code quality attributes, such as flexibility, understandability, reusability, and maintainability. It was found that, similar to the internal attributes ranking results, Extract Class was ranked in the top position by both teams. The ANP team found the ANP to be a helpful tool to guide the team toward the most important refactoring techniques, which helped reduce the time spent practicing refactoring based on the proposed criteria.

In test-driven development, the ANP was used to select the type of automated testing and to rank the release indicators. Several criteria, such as developer understanding, time, cost of execution, GUI, and test coverage, were selected to evaluate each testing type. The ANP was applied to select one of the following testing types: automated testing, semi-automated, and manual. Both teams found automated testing to be the best choice. Another use of the ANP in testing practice was ranking the release indicators. The participants evaluated three software release indicators, which are percentage of defects reported, percentage of code executed without error,
and number of test cases completed. These indicators were evaluated with respect to code quality, velocity, design, cost, and simplicity. The results showed that the percentage of execution without error was ranked in the top position by both teams. However, different projects sizes with different criteria might lead to different results.

In general, the participants found the application of ANP in the XP life cycle to be positive and suggested that it can improve decision quality, as well as increasing the level of user satisfaction. The ANP assists the development team in resolving conflicting opinions between team members based on a scientific approach, while considering everyone’s voice. However, in simple design and refactoring practices, the use of ANP was more beneficial than others. In these two areas, the effects of the ANP use were clear in terms of increasing the team performance by selecting the most suitable design tool and reducing the time spent practicing refactoring by concentrating on the most beneficial refactoring patterns. The ANP is a powerful tool for making decisions within multi-criteria environment, and it allows decision makers to consider the interdependencies among the different criteria, which helps in making accurate decisions. However, it loses its powerful with an increasing number of alternatives and criteria; therefore, in order to benefit from ANP, it is important to apply it with a small number of criteria and alternatives to avoid consuming an excessive amount of time. The limitations of this study are as follows:

- This study considers only two teams working on a small project due to the time constraint. Large projects have more teams and more communication issues, which makes it questionable if the ANP will ensure the XP team’s communication in large projects or not.

- The criteria selected in this study are not mutually restricted; this is the ANP approach’s limitation. Decision makers can define various criteria.

- The ANP is designed for a small number of criteria and alternatives; however,
for a larger number of criteria and alternatives, the Best and Worst method might be a good choice.

6.2 Future Work

In XP, there are other areas which could benefit from the application of the ANP. Especially, when practicing planning game, ANP might be applied to prioritize software requirements (user stories). Often, there are dependencies between the system requirements; therefore, the ANP can be a good choice for prioritizing user stories. Also, prioritizing user stories using ANP can be performed from two perspectives: customers’ and developers’ perspectives. Customers can prioritize user stories by considering criteria such as value and urgency, while developers prioritize user stories while considering criteria like risk, complexity, and cost. This structure will allow us to prioritize user stories by considering all team members’ opinions.

Moreover, in simple design practice, ANP can be used to prioritize CRC cards. The CRC cards offer several advantages in simple design practice, such as measuring the quality of the design and their flexibility of use during validation changes. Therefore, prioritizing CRC cards can allow XP team members to measure the software quality and guarantee the design simplicity. The ANP can be used to prioritize CRC cards based on three criteria, which are collaboration, responsibility, and stability.

In addition, the ANP can be used in test-driven development practice in order to choose the suitable test case candidates to be automated in regards to: 1) test execution repeatability, 2) user frequency, 3) test data availability, and 4) execution step accuracy.

Furthermore, the XP team can use the analytic network process in metaphor practice in order to select the best metaphor from various given systems metaphors. Describing the system in a consistent way is significant in order to ensure clear un-
derstanding from all team members, and to avoid any conflicts between the members. The XP team can select the best system metaphor based on several criteria such as vocabulary, correctness, and expandability.

Other future work that can be done is building an automated ANP tool that is intended to meet the XP process and its values. Such a tool will make ANP much more advantageous. It ought to also consider the time issue, which emerged in a portion of the XP practices and rapidly reacted to the changes and requests of the recalculation.

Also, the ANP can be integrated with different techniques in order to increase accuracy in its results; for example, the ANP can be integrated with a fuzzy set in order to provide more options in dealing with uncertain judgments. This integration will allow ANP to handle the disadvantage of dealing with the subjective judgments and roughness while evaluating the network components.

In this thesis, the ANP is introduced into the extreme programming process; however, this combination has a good potential of success if it is introduced to another agile method. For example, the ANP can be used with Scrum in order to improve the decision making process when it is needed during the Scrum development cycle. Scrum method has similarities with extreme programming; therefore, further research can investigate the application of ANP with the Scrum process.

The cost of applying the ANP decision approach or not is another aspect that needs to be investigated. Projects have certain goals and budgets, so applying the ANP as a decision approach might depend on various factors, such as comparing the benefits that it will bring to the project versus the cost of applying the ANP.

Finally, additional quantitative analysis is needed to study the effect of the ANP on cost, risk, and time. In our study, the ANP proved its ability in various areas of the XP life cycle, but further investigation, especially in industrial environments, is needed in order to test its abilities in various environments.
REFERENCES


[32] Hakyeon Lee, Sora Lee, and Yongtae Park. Selection of technology acquisi-


[85] Ellis Horowitz and Barry W Boehm. Practical strategies for developing large software systems. Addison-Wesley Reading, Ma, 1975.


[96] Sultan Alshehri and Luigi Benedicenti. Using the analytical hierarchy process as a ranking tool for user story prioritization techniques. In *Proceedings of the 8th


[114] Matthias M Muller and Walter F Tichy. Case study: extreme programming in a


