A FRAMEWORK FOR COTS SOFTWARE EVALUATION AND SELECTION FOR COTS MISMATCHES HANDLING AND NON-FUNCTIONAL REQUIREMENTS

FERAS HAMED AL-TARAWNEH

DOCTOR OF PHILOSOPHY
UNIVERSITI UTARA MALAYSIA
2014

Permission to Use

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences
UUM College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok

Abstrak

Keputusan ketika membeli perisian *Commercial Off-The-Shelf* (COTS) memerlukan garis panduan yang sistematik supaya perisian COTS yang sesuai boleh dipilih bagi menghasilkan penyelesaian yang berdaya maju dan berkesan kepada organisasi. Walau bagaimanapun, rangka kerja penilaian dan pilihan perisian COTS yang sedia ada lebih menumpukan pada aspek kefungsian dan tidak memberi perhatian yang mencukupi untuk mengendalikan ketidaksepadanan antara keperluan pengguna dan spesifikasi perisian COTS, serta tidak mengambil kira keperluan bukan kefungsian. Oleh yang demikian, satu rangka kerja baharu bagi penilaian dan pemilihan perisian COTS dalam menyelesaikan ketidaksepadanan keperluan dan mengambil kira keperluan bukan kefungsian sangat diperlukan. Justeru itu, kajian ini bertujuan untuk membangunkan rangka kerja baharu bagi penilaian dan pemilihan perisian COTS yang memberi penekanan terhadap pengendalian ketidaksepadanan keperluan dan mengambil kira keperluan bukan kefungsian. Kajian ini telah dijalankan dengan menggunakan metodologi mod campuran yang melibatkan teknik kaji selidik dan temu bual. Kajian dilaksanakan dalam empat fasa: pelaksanaan kaji selidik dan temu bual di 63 buah organisasi untuk mengenal pasti kriteria penilaian COTS, pembangunan rangka kerja perisian COTS dengan menggunakan Teori Penilaian, pembangunan teknik membuat keputusan yang baharu dengan menerapkan Proses Analisis Hierarki dan Analisis Jurang bagi mengendalikan ketidaksepadanan perisian COTS, dan pengesahan kebolehlaksanaan dan kebolehpercayaan rangka kerja Penilaian dan Pemilihan perisian COTS (COTS-ESF) yang dicadangkan dengan merujuk kepada semakan pakar, kajian kes, dan pengesahan ukur takat. Kajian ini telah mengenal pasti lima kriteria penilaian bagi perisian COTS: Kualiti, Domain, Seni Bina, Persekitaran Operasi dan Reputasi Pembekal. Ia juga menyediakan teknik membuat keputusan dan proses lengkap untuk menjalankan penilaian dan pemilihan perisian COTS. Hasil kajian menunjukkan bahawa aspek-aspek rangka kerja tersebut yang dinilai adalah sesuai dan berpotensi serta praktikal untuk digunakan dalam persekitaran sebenar. Sumbangan kajian ini merentangi kedua-dua perspektif penyelidikan dan praktikal dalam bidang penilaian perisian dengan memperbaiki proses membuat keputusan dan menyediakan garis panduan yang sistematik untuk menangani isu pembelian perisian COTS berdaya maju.

Kata kunci: Penilaian perisian *Commercial Off-The-Shelf*, Pemilihan perisian *Commercial Off-The-Shelf*, Keperluan bukan kefungsian, Pengendalian ketidaksepadanan, Teori penilaian.

Abstract

The decision to purchase Commercial Off-The-Shelf (COTS) software needs systematic guidelines so that the appropriate COTS software can be selected in order to provide a viable and effective solution to the organizations. However, the existing COTS software evaluation and selection frameworks focus more on functional aspects and do not give adequate attention to accommodate the mismatch between user requirements and COTS software specification, and also integration with non functional requirements of COTS software. Studies have identified that these two criteria are important in COTS software evaluation and selection. Therefore, this study aims to develop a new framework of COTS software evaluation and selection that focuses on handling COTS software mismatches and integrating the non-functional requirements. The study is conducted using mixed-mode methodology which involves survey and interview. The study is conducted in four main phases: a survey and interview of 63 organizations to identify COTS software evaluation criteria, development of COTS software evaluation and selection framework using Evaluation Theory, development of a new decision making technique by integrating Analytical Hierarchy Process and Gap Analysis to handle COTS software mismatches, and validation of the practicality and reliability of the proposed COTS software Evaluation and Selection Framework (COTS-ESF) using experts' review, case studies and yardstick validation. This study has developed the COTS-ESF which consists of five categories of evaluation criteria: Quality, Domain, Architecture, Operational Environment and Vendor Reputation. It also provides a decision making technique and a complete process for performing the evaluation and selection of COTS software. The result of this study shows that the evaluated aspects of the framework are feasible and demonstrate their potential and practicality to be applied in the real environment. The contribution of this study straddles both the research and practical perspectives of software evaluation by improving decision making and providing a systematic guidelines for handling issue in purchasing viable COTS software.

Keywords: Commercial Off-The-Shelf evaluation, Commercial Off-The-Shelf selection, Non-functional requirements, Mismatches handling, Evaluation theory.

Acknowledgment

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

First and foremost all praise and thanks go to Allah for giving me the strength and patience, and providing me the knowledge to accomplish this research study.

In this occasion I would like to express my gratitude to a number of people whose admission, permission, and assistance contribute to finish my long story with PhD.

I would like to express my sincerest thanks and deepest gratitude to my supervisor Assoc. Prof. Dr. Fauziah Baharom, for her excellent guidance, caring, patience, providing me with an excellent atmosphere for doing research, and sharing of all her research experiences throughout these challenging years. I would like also to thank my second supervisor Assoc. Prof. Dr. Jamaiah Hj Yahaya for her continuous guidance, fruitful feedback, and moral support.

My highly appreciation to Assoc. Prof. Dr. Faudziah Ahmad and Assoc. Prof. Dr. Haslina Mohd for the useful comments and suggestions to improve my thesis. My sincere thanks must also go to the members of viva committee: Assoc. Prof. Dr. Huda Hj Ibrahim as chairman, Prof. Dr. Siti Salwah Salim from University of Malaya (UM) as external examiner, and Dr. Nor Laily Hashim as internal examiner.

On the personal level, I would also like to express my gratitude to my parents and my beloved family members for patience and support throughout my five years plus of difficult endeavor. I guess they are the most who suffered throughout this period. My gratitude also goes to all my colleagues in the PhD journey, specifically for the discussions and sometimes the heated arguments on the better ways to perform my research.

Thank You All Very Much

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

COTS	Commercial-Off-The-Shelf
CBS	COTS-Based Systems
CBD	COTS-Based systems Development
OTSO	Off-The-Shelf Option
PORE	Procurement-Oriented Requirements Engineering
CSSP	COTS Software Selection Process
CAP	COTS software Acquisition Process
PAREMO	Balanced Reuse Model
MiHOS	Mismatch handling aware COTS Selection
CRE	COTS-based Requirements Engineering
STACE	Social-Technical Approach to COTS Evaluation
DC	Developing Country
GUI	Graphic User Interface
IDC	International Data Corporation
CBA	COTS-based Application
USC	University of Southern California
TT&C	Telemetry, Tracking, and Control
CERES	Center for Research Support
WSM	Weighting Scoring Method
AHP	Analytical Hierarchy Process
IusWare	IUStitiasoftWAR
CISD	COTS-based Integrated System Development
PRISM	Portable, Reusable, Integrated Software Model
CEP	Comparative Evaluation Process
CF	Confidence Factor
IESE	Institute for Experimental Software Engineering
RCPER	Requirements-driven COTS Product Evaluation Process
CARE	COTS-Aware Requirements Engineering

PECA	Plan, Establish, Collect, and Analyze
CSCC	Combined Selection of COTS Components
GCS	General COTS Selection
unHOS	uncertainty Handling in COTS Selection
GQM	Goal Question Metrics
ISO	International Organization for Standardization
QFD	Quality Function Deployment
BBN	Bayesian Belief Network
SPA	Software Process Assessment
MCDM	Multi-Criteria Decision Making
ERP	Enterprise Resource Planning
KM	Knowledge Management
API	Application Programming Interface
CAR/SA	COTS-Aware Requirements and Software Architecture
NFR	Non-Functional Requirements
IRC	Identifying mismatches Resolution Constraints
CRC	Considered Resolution Constraint
SDMP	Systematic Decision Making Process
C-QM	COTS-Quality Model
ISO/IEC	International Organization for Standardization and international Electro technical Commission
UK	Unite Kingdom
SME	Small Medium Enterprise
SE	Software Engineering
SPSS	Software Package for Social Sciences
JAD	Joint Application design
ASP.NET	Active Server Pages.Net
VB.Net	Visual Basic.Net
IT	Information Technology
SD	Standard Deviation

CEC	COTS Evaluation Criteria
ANC	Average Normalized Column
CR	Consistency Ratio
CI	Consistency Index
RI	Random Index
FMML	Final Mismatching Level
ML	Matching Level
MML	Mismatching Level
FFS	Final Fitness Score
WBS	Work Breakdown Structure
SLA	Service Level Agreement
DM-PT	Decision Making- Prototyping Tool
ID	Identification number
COB	College Of Business
CAS	College of Art and Science
CLGIS	College of Low, Government, and International Studies
PCs	Personal Computers
SMIS	Student Management Information System
OSS	Open Source System
OTS	Off-The-Shelf

CHAPTER ONE
INTRODUCTION

1.1 Introduction

This chapter provides an introduction to the field of this research by describing the background of the study and discussing the research problem. The research questions are then presented and used to construct the research objectives. Finally, the chapter describes the scope of this research; as well as highlighting the significance of the research. The chapter concludes with an overview of the remaining chapters of this thesis.

1.2 Background

The world of software development has significantly evolved from development-centric to a procurement-centric approach. In other words, this new approach has been introduced as an alternative software development approach which focused on building systems through pre-packaged solutions assembling, usually known as Commercial-Off-The-Shelf (COTS) software, and migrating existing systems towards COTS-Based Systems (CBS) (Gupta et al., 2012). Nowadays, most organizations have decided to change from in-house development towards COTS software integration in order to reduce the maintenance cost, development time, and operating, testing, and validating efforts (Couts & Gerdes, 2010). Thus, COTS software has become strategic and economic way for building large and complex systems.

COTS software is a term referred to a piece of software that is developed and supported by outside suppliers (so-called vendors) to provide additional functionalities within a final system (Torchiano & Morisio, 2004). More specifically, this term is defined by the Software Engineering Institute as " *a software product that is: (1) sold, leased, or licensed to the general public; (2) offered by a vendor trying to profit from it; (3) supported and evolved by the vendor, who retains the intellectual property rights; (4) available in multiple, identical copies; and (5) used without source code modification by a consumer*" (Morisio & Torchiano, 2002; Yanes et al., 2012). In addition, Galorath (2005) defines COTS software as retail software that is considered to be a compiler, a software tool, and an operating system. CBS is developed based on the selection, adaptation, and integration of one or more COTS software. This process is called COTS-Based system Development (CBD) (Kvale at el., 2005).

Using the COTS software changes the way of building systems from scratch or in-house building to assembly pre-existing COTS software, which has been tested several times by many other customers. Thus, this approach grants the opportunity to lower the development costs by sharing them with other customers, to provide rapid delivery to end users, and to reduce the development times and efforts. Apart from that, CBS also enhances the reliability, flexibility, and reusability of the systems (Bertoa & Vallecillo, 2002; Ibrahim et al., 2011; Wanyama & Far, 2008).

Consequently, the increased demands on the COTS software in the last decades have flooded the software market with a huge numbers of COTS software. Therefore,

selecting the most suitable COTS software has become the main challenge to the organizations that intend to use such software. This challenge occurs when there are many similar COTS software are available in the market produced by different vendors with different capabilities and qualities (Gupta et al., 2012; Wanyama & Far, 2008). As COTS software evaluation and selection provides the opportunity to select one or more of these software that can fulfill most users' requirements (Ibrahim et al., 2009; Kunda, 2002; Mohamed et al., 2008), any wrong decision during this process will lead to catastrophic results. In other words, the wrong decisions will reflect negatively on the organization entirely by increasing the cost, time, and effort, and will eventually affect the performance and quality of the final system (Falessi et al., 2011; Goswami & Acharya, 2009; Ibrahim et al., 2011; Lin et al., 2007). Therefore, the evaluation and selection process is considered as a critical and important process because the successful of the final system implementation depends largely on it (Ibrahim et al., 2009; Mohamed at el., 2008; Pande, 2012).

Several problems that are related to the COTS software evaluation and selection indirectly become a challenge to the organizations that aim to use the COTS software (Alves et al., 2001; Jha & Bali, 2012). One of the problems is that the COTS software has been rapidly changed and evolved which makes the evaluation process more difficult. This is because a new release of the COTS software emerges new desirable features that are not available in the COTS software that is currently being evaluated (Ayala, 2008; Bhuta & Boehm, 2007; Jingyue et al., 2009; Ulkuniemi & Seppänen, 2004). The "Black box" nature of the COTS software will negatively affect on its evaluation since the internal source code is not available. In

other words, the functions of the COTS software and their behaviors are only available for the users (Alves et al., 2001; Javed et al., 2012; Kumare & Singh, 2012; Yang et al., 2011).

1.3 Research Problem

Most of the proposed methods of addressing the COTS software evaluation and selection problems, such as the Off-The-Shelf Option (OTSO) and Procurement-Oriented Requirements Engineering (PORE), are not considered as practical methods because these methods are theoretical, manual, and labour intensive (Pande, 2012; Wanyama & Far, 2008). As a result, majority of organizations prefer to select and evaluate the COTS software using an ad-hoc manner (Couts & Gerdes, 2010; Gupta et al., 2012; Jingyue et al., 2009; Navarrete at el., 2007; Neubauer & Stummer, 2007; Pande, 2012; Wanyama & Far, 2008).

Among the ad-hoc manner basis used by the organizations in evaluating and selecting the COTS software are by depending on the experiences of development team or by following their intuition. The relationships between the organizations and particular vendors may also lead to the selection of the COTS software regardless of its capability and quality (Jingyue et al., 2009; Kunda, 2003; Alves et al., 2001; Alves et al., 2000). The lack of systematic, repeatable, and well-defined process for the COTS software evaluation and selection in the industry will keep the organizations under pressure (Alves et al., 2001). For instance, the development team that lack of experiences might not provide a thorough plan for the selection

process. As a result, the decision may become inaccurate and more risky, and learning from the previous cases is difficult (Kunda, 2003; Alves et al., 2000).

Moreover, many studies such as Pande (2012), Gupta et al. (2012), Jadhav and Sonar (2009), Li et al. (2009), Ayala et al. (2011), and Torchiano and Morisio (2004) show that there is a gap between theory and practice on the COTS software evaluation and selection methods whereby many practitioners are still rely on the ad-hoc manner in evaluating and selecting the COTS software. This gap exists because there are main issues and problems that have not been addressed in the previous methods which include: i) lack of addressing the mismatches between user's requirements and features of the COTS software, and ii) lack of handling non-functional requirements of the COTS software (Kiv et al., 2010; Javed et al., 2012).

1.3.1 COTS Software Mismatches

Identifying the mismatches between the COTS software and customer's requirements has an important role in supporting the decision making relating to the COTS software selection process and in reducing the time, effort, and costs spent over the COTS software adaptation and integration phases in CBD (Alves et al., 2005; Guerra et al., 2004; Ibrahim et al., 2011; Mohamed et al., 2011). The COTS software mismatches are defined as shortages or excesses of COTS software features against user's requirements. The mismatches occur because the COTS software is usually developed for general usage, while user's requirements are determined based on their contexts (Alves & Finkelstein, 2003; Ibrahim et al., 2009).

A few studies have been conducted to address the issue of COTS software mismatches. Among the studies are those conducted by Alves et al. (2005) and Mohamed et al. (2008). Both studies proposed methods of identifying and classifying the COTS mismatches. On the other hand, Kiv et al. (2010) used the goal-driven agent-oriented approach to address the COTS mismatches issue. Even though their study have a lack to provide accurate method for calculating the COTS mismatches and did not include the non-functional requirements when addressing the mismatches, the researchers did stress on the importance of early identification, better understanding and estimation of the COTS mismatches will provide valuable insight on the decision of the COTS software selection. This helps to reduce any related risk of system development failure. Furthermore, by handling the COTS software mismatches early during COTS software evaluation and selection, the proceeding activities in CBD such as adaptation and integration can be conducted easily and efficiently particularly in terms of time and cost.

From their review, Javed et al. (2012) and Sarkar (2012) mentioned that most of the existing methods, such as OTSO, COTS Software Selection Process (CSSP), COTS software Acquisition Process (CAP), and Balanced Reuse Model (PAREMO), tend to neglect the mismatches between the COTS software and user's requirements especially on the non-functional requirements. They have also neglected to suggest ways of identifying and solving the mismatches problem in order to support and enhance decision making in the selection process. This result is also supported by (Alves & Finkelstein, 2003; Alves et al., 2005; Kiv et al., 2010; Kvale et al., 2005; Mohamed et al., 2011; Mohamed at el., 2008; Mohamed at el., 2007; and Ye &

Kelly, 2004). However, a few of methods, such as PORE and Mismatch-Handling aware COTS Selection (MiHOS), have attempted to address COTS software mismatches problem but they did not consider the COTS mismatches constraints such as the cost and the risks of the potential mismatches' solution which reflect negatively on the decision making process.

1.3.2 Non-Functional Requirements

Many studies have analyzed and compared the existing COTS software evaluation and selection methods such as those carried out by Ayala (2008), Jeved et al. (2012), and Sarkar (2012). All of these studies stressed that the existing COTS software evaluation and selection methods are focusing on the functional requirements instead of the non-functional requirements. The non-functional requirements refer to the overall characteristics or attributes of a system such as the quality, vendor, and organization (Glinz, 2007; Pavlovski & Zou, 2008; Sommerville, 2011). Similarly, the work performed by Beus-Dukic (2000), Crnkovic et al. (2005), and Kaur and Mann (2010) indicate that the COTS software evaluation needs to address several kinds of non-functional requirements such as architectural, domain, and user organization in order to provide all required evaluation criteria that are needed for completing the COTS evaluation process.

Although these non-functional requirements are difficult to gather, express, qualitative, and test, they play an important role in distinguishing the similarities of the COTS software alternatives and in improving the understanding of the COTS software features through the evaluation process (Beus-Dukic, 2000). In addition,

Sommerville (2011) mentions that the non-functional requirements are the complete characteristics of the software rather than the individual features, which might represent a deciding factor on the survival of software.

Other existing methods and models for the COTS software evaluation and selection are MiHOS, DesCOTS, COTS-based Requirements Engineering (CRE), Social-Technical Approach to COTS Evaluation (STACE) and CAP, which stress on different requirements. For example, MiHOS evaluate and select the COTS software based on the functional requirements, while DesCOTS focuses on the quality requirements. Although the CRE focuses on the requirements engineering and relies on NFR framework for representing the identified non-functional requirements, it does not address several kinds of non-functional requirements such as the domain, and vendor characteristics. The STACE and CAP attempt to handle the non-functional requirements by proposing some of them in a high-level and simple way by not taking into consideration of all the required attributes of the non-functional requirement (Alves & Castro, 2001; Ayala, 2008; Beus-Dukic, 2000; Ye & Kelly, 2004).

The lack of careful consideration of the non-functional requirements increases the risk of failure of the COTS software evaluation and selection process. For instance, the cost of the final system might be in jeopardy because this kind of requirements corresponds to the strategic and business objectives of the organization as a whole. Thus, the non-functional requirements should be considered explicitly as decisive

criteria for evaluating and selecting the appropriate COTS software (Ayala, 2008; Cortellessa et al., 2007; Minkiewicz, 2005; Pande, 2012).

As discussed in sub-sections 1.3.1 and 1.3.2, due to the lack of existing COTS selection methods in providing a viable and effective solution to the organizations in the industry, these organizations are still in need of a systematic, effective, and well-defined process for evaluating and selecting the COTS software. Therefore, this research aims to develop a new framework that addresses the limitations and problems in the current selection COTS methods which are: the COTS software mismatches problem and non-functional requirements problem.

1.4 Research Questions

The research questions of this study have be constructed as:

1. What are the important processes, activities, and techniques involved in the COTS software evaluation and selection?

2. What are the non-functional requirements that are required for supporting the COTS software evaluation and selection?

3. How to address the mismatches between the features of COTS software and non-functional requirements through the evaluation and selection process?

4. How to evaluate and select the most suitable COTS software components in a systematic approach?

5. How to evaluate the applicability of the new framework in the industry?

1.5 Research Objectives

The aim of this research is:

"*To propose a new framework for COTS software evaluation and selection based on handling COTS mismatches and non-functional requirements.*"

In order to achieve this research aim, several specific research objectives have been formulated:

a) To identify the non-functional requirements and establish the common evaluation criteria required to support the COTS software evaluation and selection.

b) To construct a new method for the handling mismatches in the COTS software and user requirements.

c) To construct a new framework for the COTS software evaluation and selection using the evaluation theory.

d) To evaluate the applicability of the proposed framework using case study approach.

1.6 Research Scope

The overall objective of this research is to develop a framework for COTS software evaluation and selection. The framework will be constructed by referring to the evaluation theory because it provides the six basic evaluation components for any evaluation process. This framework focuses on addressing the lack of handling the non-functional requirements by identifying the technical requirements such as quality and architectural requirements, and the non-technical requirements such as

vendor and user organization requirements. The framework also handles the COTS software mismatches problems through addressing the Multi Criteria Decision Making (MCDM) and Gap Analysis (GA) techniques. Since the COTS software evaluation and selection is considered as the decision making process, the MCDM is used as it provides appropriate techniques, such as AHP technique, to select the fitness among several alternatives based on a set of criteria. The GA was also chosen since it is the most suitable technique for handling the gaps or mismatches problem.

In order to keep focused, this research only concentrates on the COTS software evaluation and selection process and leaves out the adaptation and integration processes. Furthermore, this research is only concerned with the selection of a single COTS software that fulfills most of the user's requirements. The target type of CBS in this research is the intermediate systems which focus on selecting one COTS software product that meets most of user's requirements. Jordan, as a developing country (DC), has been chosen in this research because its organizations have converted their manual system using computer technology, specifically the COTS software technology in order to increase the productivity and quality of the system with lower cost. In the validation stage, Malaysia and Jordan have been chosen to validate the framework effectiveness and applicability in the real life. Both Malaysia and Jordan represent two different environments of selecting two different kinds of COTS software. Therefore, the feasibility of the framework in these countries proved that the framework can be applicable in other countries for different COTS software.

1.7 Significance of Research

This research will be significance endeavor in promoting the evaluation and selection process to support the stakeholders in practice when using COTS software, as well as it will be helpful and support the body of knowledge in several fields. This significance is described as follows:

Body of knowledge

The high-level goal of this research is to propose a new framework for the COTS software evaluation and selection. Therefore, this research contributes towards the field of software engineering, particularly in the area of software evaluation, the non-functional requirements by providing set of evaluation criteria, and the decision making process by addressing the mismatches problem.

Benefit to the stakeholders

The new framework proposed by this research will be very helpful to organizations in selecting the most suitable COTS software in a short time and effort; to the organizations' evaluators and decision makers, the selection process can be carried out in a more systematic, accurate, and smooth way through the set of well-defined processes, activities, evaluation criteria, and techniques. In addition, the main result of this framework, such the mismatches information, will be very useful and beneficial as a feedback for the vendors to improve their COTS products. Finally, the proposed evaluation criteria and decision making technique will be helpful and beneficial to the academic researchers especially in encouraging to improve, extend, and adapt these components in the software evaluation field or in other fields.

1.8 Thesis Organization

This research consists of seven chapters, including this chapter. The remaining chapters are organized as follows:

Chapter Two: Literature Review

This chapter presents a critical analysis on the COTS software evaluation and selection. The focus gives on existing methods and models for COTS software evaluation and selection, the main processes, and techniques. It also presents an overview of the efforts made to address the evaluation criteria, and COTS software mismatches problem as well as COTS software evaluation and selection critical issues.

Chapter Three: Research Methodology

This chapter firstly discusses the evaluation theory that presents the six required evaluation components for developing the software evaluation framework. Next, the research methodology that was used to carry out the research and to achieve the research objectives was presented by discussing in detail the four phases to develop and evaluate the new framework for supporting the COTS software evaluation and selection in industry.

Chapter Four: Empirical Study

This chapter presents the findings of the empirical study that was conducted in Jordan. The findings that are related to the current problems, important processes, techniques, and evaluation criteria were presented and discussed. This chapter also

provides the understanding about the current practice of CBS development and the process of the COTS selection in real life.

Chapter Five: Framework Development

In this chapter, the framework that was developed based on the evaluation theory components is presented and discussed in detail. The components include the target of the evaluation, the evaluation criteria which represented by the COTS evaluation criteria (CEC), the yardstick, data gathering techniques, synthesis technique (decision making technique). These components interact through a set of evaluation processes.

Chapter Six: Framework Evaluation

This chapter presents the two stages of the research process: verification and validation stages. The verification stage was conducted using the experts review approach coupled with Delphi technique. In the framework validation stage, the supported software tool and its structure and components are presented. The case study and yardstick approaches were used to evaluate the validity and applicability of the proposed framework.

Chapter Seven: Conclusion and Future Work

In this chapter, the conclusion of the research is presented through review of the studies that have been conducted to achieve the research objectives. It also highlights the contributions of this research, and finally a set of issues is presented as a future research at the end of this chapter.

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

This chapter aims to describe the state-of-the-art concerning the COTS selection issues. As a start, the chapter provides an overview of the COTS software by focusing on the evaluation and selection process particularly on the existing methods, MCDM in supporting the COTS software selection, the handling of COTS mismatches, and common processes and activities. The discussion continues by highlighting related studies on the evaluation criteria and existing empirical studies. Finally, the chapter is concluded by presenting the main issues and challenges in COTS software evaluation and selection and the chapter summary.

2.2 Overview of the COTS Software

In general, almost all retail software applies the use of Commercial Off-The-Shelf (COTS) term (Bachmeyer, 2004). This kind of software is usually represented by various definitions such as product that: i) is sold, leased, or licensed to the public; ii) does not provide internal code for user; and iii) is supported by vendor who controls its development and has nontrivial installed base (Basili & Boehm, 2001; Brownsword et al., 2000).

In general, the COTS software is an independent software designed, constructed, tested, documented, and upgraded by vendors to provide desired functionalities and increase productivity (Torchiano & Morisio, 2004; Sommerville, 2001). Among the

examples of the COTS software are operating systems (e.g. windows), database systems, case tools (e.g. editors, compilers, GUI builders, & test tools), and utilities (e.g. word processor, spreadsheet) (Skramstad, 2005).

The use of the COTS software is widely spread and still expanding in the world of systems development (Gayen & Misra, 2009; Sommerville, 2004). The results from the survey conducted by International Data Corporation (IDC) showed that there are more than 50% of the systems developer organizations around the world that used the COTS software for building most of their recent projects, and this percentage is still rising continuously (Li et al., 2009). Another study done by the Gartner Group indicates that the COTS software is anticipated to represent over 90% of the software applications running in major corporations (Martínez, 2008; Megas et al., 2013; Leontie et al., 2009). In addition, the latest estimating of the annual market of the COTS software has reached at almost $200 billion that make this market worldwide (Keil & Tiwana, 2005). Figure 2.1 presents the increment of 28 % in 1997 to 70 % in 2002 of the use of the COTS-Based Application (CBA) in e-service projects by the University of Southern California (USC).

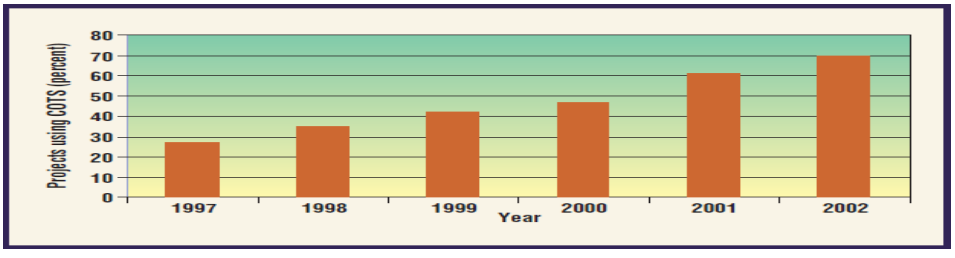


Figure 2.1. The COTS-based Applications' Growth in USC E-Service Projects (Yang et al., 2005)

Since the software industry has moved towards extensive use of the COTS software, software engineers have becoming more aware of the importance of such technology especially in improving applications development with less cost and time, offering rapid advances of technology, and increasing standardization of technology (Simmons & Dillon, 2006; Ulkuniemi & Seppanen, 2004; Callagy, 2007). Unfortunately, the software engineers usually do not have access to the source code of the COTS software. Moreover, there are limited descriptions of the functionalities and qualities, which may require substantial effort for inspecting and capturing the required information before deciding, accepting, or rejecting such software. Table 2.1 summarizes a number of advantages and disadvantages of the COTS software.

Table 2.1

The COTS Software Advantages and Disadvantages.

		Explanation	Sources
Advantages	Increased productivity	Reliability, availability, and flexibility	Yang et al., 2005; Alves et al., 2000; Turkosy et al., 2012; Johar & Goel, 2011
	Localization	Focusing on the effect of changes on particular section of application	Sedigh-Ali et al., 2002; Torchiano et al., 2002
	Increased competition	Up to date, rapid delivery, and high quality	Albert & Brownsword, 2002, Callagy, 2007; Kesseler, 2008; Fröberg, 2000
	Lowering cost	Reducing the coding, documented, and testing costs	Kesseler, 2008; Sommerville, 2011, Charpentier & Salois, 2000, Fang et al., 2010
	Reduced time-to-market	Getting these components to the marketplace faster and cheaper	Kesseler, 2008; Fang et al., 2010; Fröberg, 2000; Li, 2006; Johar & Goel, 2011
	Affordability	More solutions are available in the market	Kesseler, 2008; Mili et al., 2000; Boehm & Abts, 1999
	Standardized functionalities	Software can be used by many organizations with the same domain.	Kesseler, 2008; Torchiano et al., 2002; Fröberg, 2000; Johar & Goel, 2011
	Developed by expert developer	The COTS vendor is an expert in the specific field of functionality.	Fröberg, 2000; Torchiano et al., 2002; Jadhav & Sonar, 2009

Disadvantages	Incompatibility	The COTS software some time fails to interact with other components in the system.	Gupta et al., 2012; Johar & Goel, 2011; Sedigh-Ali et al., 2002
	COTS upgrading	Creating various risks such as relying on vendors, not meeting user requirements, or leading to mismatches and side effects.	Boehm & Abts, 1999; Basili & Boehm, 2001; Li, 2006; Wile et al., 2010
	Vendor goes out of business	If vendor discontinuous a COTS software, the upgrading, maintenance, and COTS supporting may also disappear.	Couts & Gerdes, 2010; Merola, 2006; Hopkins, 2000
	Lack in documentation	Document is not available, incomplete, or in most cases is not reliable.	Mujeeb-u-Rehman et al., 2005; Wile et al., 2010; Ibrahim et al., 2011; Boehm & Abts, 1999
	Black box testing	Without internal source, evaluation is difficult since testing can only be done on the COTS functionality and behaviour	Sedig-Ali et al., 2002 Li, 2006; Yang et al., 2011
	"Vapourware"	Vendors provide some unimplemented functionalities.	Mujeeb-u-Rehman et al., 2005
	One way communication with vendor	Many questions from users are not responded by vendors as well as the legal implications.	Morisio et al., 2002; Sankaran et al., 2011

The overall quality of a final COTS-Based System (CBS) often reflects the quality of the COTS software products (Alvaro et al., 2010; Javed et al., 2012). In the last decade, the topic of CBS has received great attention from the research and industrial communities (Kumari & Upadhyaya, 2011; Torchiano & Morisio, 2004) because the system is developed based on the composition and integration of one or more COTS software (Brownsword et al., 2000; Egyed & Balzer, 2006; Morisio et al., 2002). Among those communities are the NASA Jet Propulsion Laboratory with its Deep Space Network Program and the Center for Research Support (CERES) that developed the Air Force Space and Missile System Center's telemetry, tracking, and control (TT&C) system (Kontio, 2008).

According to Carney (1997), Comella-Dorda et al. (2004), Torchiano (2001), and Wallnau et al. (1998), the CBS can be divided into three classifications. The first

category is the **COTS-Solution Systems (Turnkey System)**, which are developed based on one or a suite of COTS software that satisfy most of the desired functionalities such as Microsoft office. The second is the **Intermediate System** that is built around one substantial COTS software (e.g. Oracle), which provides the main functions in the system that may require a customization to meet user's needs. Finally, the third is the **COTS-Aggregate Systems** that is built based on the integration of various COTS software components of which have the same level of importance (Kumar & Suri, 2012; Li et al., 2006).

As for this research, the focus is on the intermediate system, as it requires selecting single COTS software that provides most of the users' requirements. The following section addresses in detail the COTS software evaluation and selection its related issues.

2.3 COTS Software Evaluation and Selection

The COTS software evaluation and selection refers to the process of identifying and determining the fitness of the software for the purpose of integrating it into a system or applying it into a specific domain (Comella-Dorda, 2004; Mohamed et al., 2007). Therefore, the successful of the COTS software evaluation and selection process is crucial in supporting the successful of the CBD (Basili & Boehm, 2001; Bhuta & Boehm, 2007; Javed et al., 2012; Vitharana et al., 2003).

In general, the selection of suitable COTS software is often considered as a non-trivial task because of the difficulty in selecting from various similar software in the

market with different qualities and capabilities (Wanyama & Lumala, 2007). Inappropriate COTS selection decision will not only reflect on the quality of the final system but also might introduce a number of risks to the organization such as:

1. Failure to support business process in an organization which will raise the issue of users' dissatisfaction (Falessi et al., 2011; Ibrahim et al., 2011; Jadhav & Sonar, 2009).

2. The losses and complexity of some daily processes which will impact an organization's financial performance (Wanyama & Far, 2008).

3. The increasing of the required time, efforts, and costs for COTS software integration and adaptation (Ibrahim et al., 2011; Jadhav & Sanor, 2009; Soni & Kodali, 2010).

The COTS software evaluation and selection is also associated with uncertainty because it involves complex decision making, multiple stakeholders, and multiple purposes (Ibrahim et al., 2011; Wanyama & Far, 2008). Therefore, many researchers have proposed several methods of addressing the COTS software evaluation and selection problem. These methods are discussed in the following section.

2.3.1 Existing Methods for COTS Software Evaluation and Selection

In this section, the review and comparisons of existing COTS software evaluation and selection methods were conducted to trace the contribution of each method contributed to the current COTS selection practice and to identify other missing issues.

According to Javed et al. (2012), and George et al. (2008), the widespread and useful methods served as a basis for other methods. These methods include the Off-The-Shelf-Option (OTSO) by Kontio (1995), Social-Technical Approach to COTS software Evaluation (STACE) by Kunda and Brooks (1999); and Procurement-Oriented Requirements Engineering (PORE) by Nucbe and Maiden (1999). Each of these methods is presented in detail while the others methods are briefly explained in the proceeding sections.

2.3.1.1 Off-The-Shelf-Option framework (OTSO)

The OTSO was the first method introduced in 1995 to facilitate a systematic, iterative, and requirement-driven COTS selection process. Typically, this method is considered as an important milestone and a foundation model for the COTS software selection methods (Mohamed at el., 2007). As shown in Figure 2.2, the OTSO consists of the five processes: evaluation criteria definition, searching, screening, evaluation, and result analysis.

Besides providing the candidates for cost-benefits estimation, the OTSO also uses the Weighting Scoring Method (WSM) for decision making purposes (Kontio, 1996).

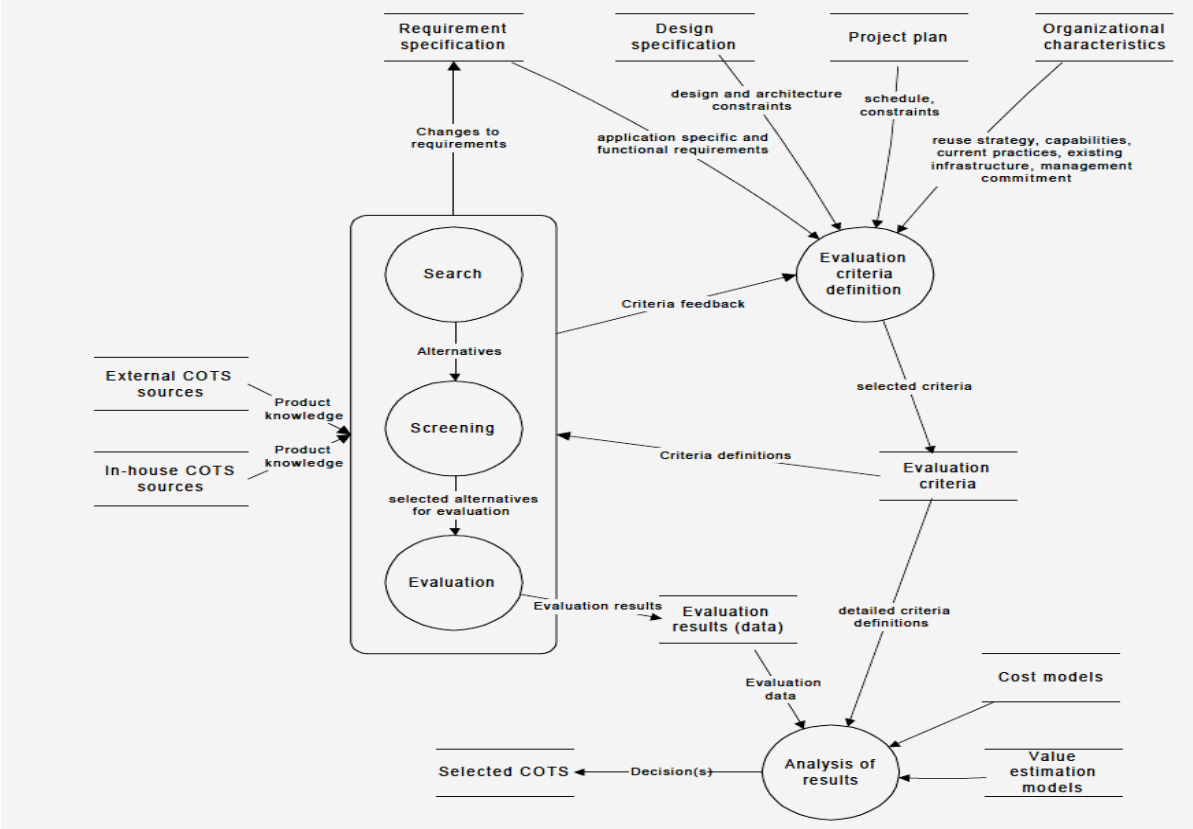


Figure 2.2. The OTSO Processes

(Kontio, 1996)

However, there are several limitations of the OTSO method. Firstly, it focuses on the function and cost rather than the non-functional requirements (i.e. user organization) (Alghamdi, 2007). In addition, it depends on the experience of personnel rather than providing specific technique of handling the extra or unrequired features (Alves & Finkelstein, 2003; Javed et al., 2012). Furthermore, the OTSO relies on the WSM technique to provide the decision making even though this technique is not efficient in handling a large number of evaluation criteria comparisons. It also ignores the

COTS software features and user's requirements mismatches (Javed et al., 2012; Ncube & Dean, 2002).

Therefore, many attempts have been made to address these limitations such as Kunda and Brooks (1999) proposed set of social-technical criteria to address the lack of non-functional requirements, while each of Alves at al. (2005), Chung et al. (2001), and Mohamed et al. (2008) provided several methods and techniques to deal with the COTS software features and user's requirements mismatches. Nevertheless, the OTSO provides explicitly definitions of the activities involved in the software components selection, which are described in form of generic model that can be used as a basic model in the software evaluation and selection process. In addition, it provides a hierarchy structure for evaluation criteria which serves as a template for situation specific criteria definition.

2.3.1.2 Procurement-Oriented Requirements Engineering (PORE)

Ncube and Maiden (1999) proposed the PORE method to address the lack of requirements engineering methods by providing iterative processes for eliciting, describing, and analyzing user requirements together with eliciting and describing the COTS software alternatives (Ncube & Maiden, 1999). In general, the PORE method consists of three components: process model, method box, and product model. The process model determines four goals while performing requirements acquisition and COTS software selection iteratively. Figure 2.3 highlights the high-level processes of the PORE method as well as ways of achieving each of the essential goals (Maiden & Ncube, 1998; Ncube & Maiden, 1999).

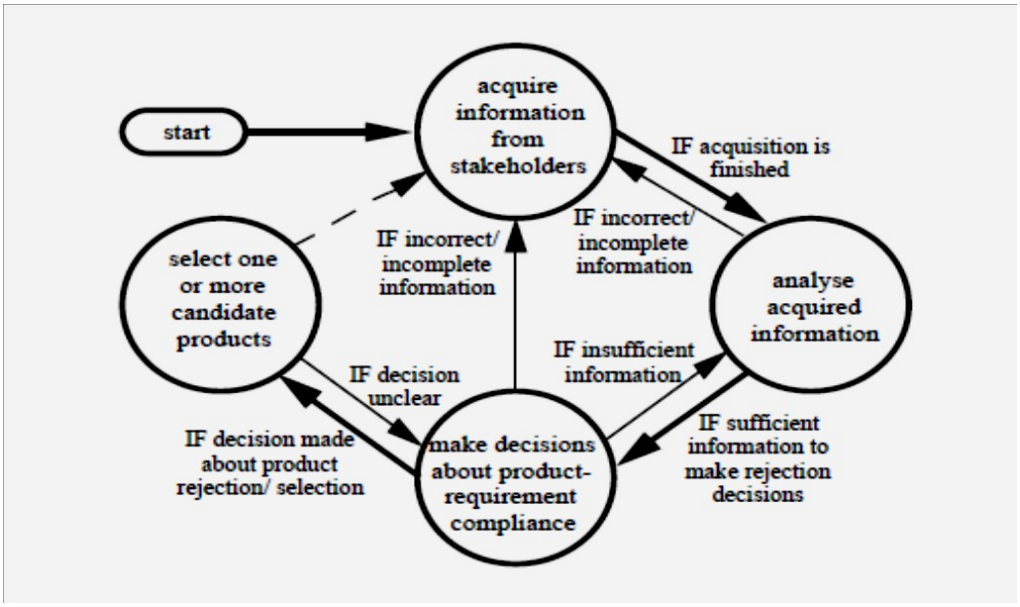


Figure 2.3. The PORE's High-level Generic

(Maiden & Ncube, 1998)

The PORE method box consists of various methods, techniques, and tools to support achieving the goals of the process model such as the AHP technique (for supporting decision making process), and card sorting and laddering (for acquiring the information about the COTS software and users' requirements) (Ncube & Maiden, 1999).

Even though it has a strong contribution to support the COTS software evaluation and selection process, the PORE method still lacking in handling clearly the mismatches between the COTS software features and the system requirements and how to eliminate those mismatches (Mohamed et al., 2007). In addition, the PORE method is depending on templates to acquire and evaluate the COTS alternatives even though these templates only provide initial view of systematic evaluation steps (Alves & Finkelstein, 2003). The iterative requirements acquisition and the COTS

software selection process become more complex as the number of requirements and the COTS alternatives increases, which requires more time and effort (Ncube & Maiden, 1999).

In this regards, many methods have been proposed to handle the PORE's limitations. For example, the MiHOS (Mohamed et al., 2008) and UnHOS (Ibrahim et al., 2011) approaches have been proposed to deal with the potential mismatches between the COTS software and user's requirements. The Plan, Establish, Collect, and Analyze (PECA) (Comello-dorda et al., 2002) also addressed the PORE' templates problem by providing high level processes and guidelines for tailoring them in different situations. Even so, the iterative process between the requirements acquisition and COTS alternatives identification and the evaluation templates, which proposed by PORE, offers a preliminary view of the required further steps to improve the process of the COTS software evaluation and selection.

2.3.1.3 Social-Technical Approach to COTS software Evaluation (STACE)

The STACE framework was developed to address the lack of attention on the non-technical or "soft" issues of the COTS software such as organizational and cost issues (Kunda & Brooks, 1999). The method was influenced by the OTSO and PORE methods in carrying out its processes.

As shown in Figure 2.4, STACE consists of the following processes: requirements definition; social-technical criteria definition; COTS software alternatives identification; and evaluation (assessment). The main feature of STACE is that it

supports user's participation throughout the evaluation and selection process. This is important because the people instead of the technical issues often cause most of the failures of the software system. Finally, it supports estimating the underlying technology before selecting the COTS software (Kunda & Brooks, 1999).

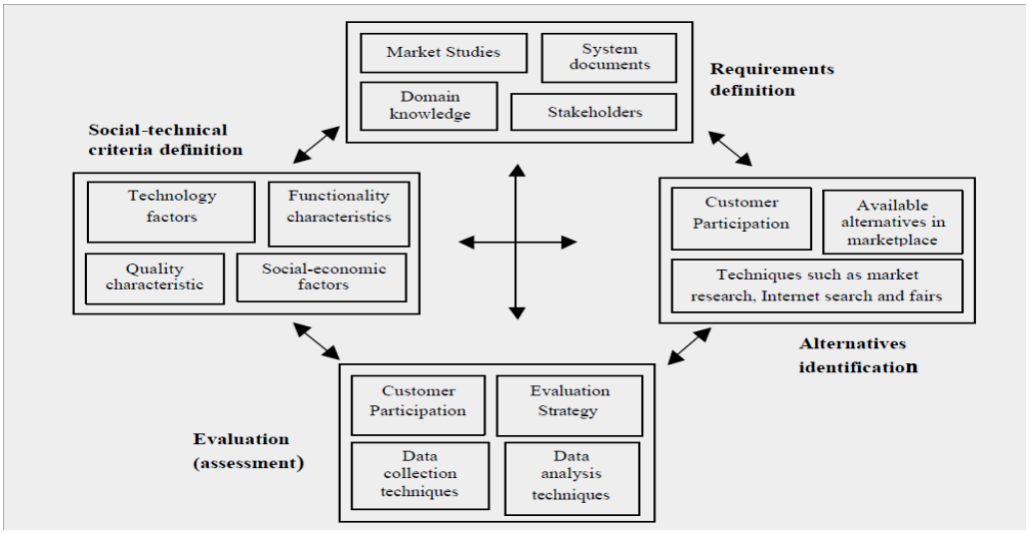


Figure 2.4. The Processes Model of the STACE Method
(Kunda & Brooks, 1999)

The limitations of the STACE method include the inability to define the requirements acquisition and specification process clearly, handle the mismatches issue, come up with resolution actions, and provide or use systematic analysis during the alternatives assessment (Alghamdi, 2007; Alves et al., 2001). Even though, the STACE approach gives emphasis to social and organizational issues related to the COTS software evaluation and selection process. Moreover, it supports the users participating in the evaluation and decision making process.

2.3.1.4 Other Existing Methods for COTS Software Evaluation and Selection

Many other models have been developed based on or as an extension of previous models (Land et al., 2008). These models contribute to the improvement of the three levels of the COTS software evaluation and selection process, which are the processes level (enhancing the current processes), evaluation criteria level (new factors), and new tools level (Jadhav & Sonar, 2009). Table 2.2 presents brief descriptions of these models which are provided emphasizing on the main ideas and tracking the enhancement made to the overall COTS software evaluation and selection.

Table 2.2

The Existing COTS Software Evaluation and Selection Methods

#	The Method	Description	Resource
1	IUStitiasoftWAR (IusWare)	It attempted to formalize the COTS evaluation and selection process, and address quality attributes.	(Morisio & Tsoukis, 1997)
2	COTS-based Integrated System Development (CISD)	It emphasized on selecting multiple fitness of the COTS software.	(Tran et al., 1997)
3	Portable, Reusable, Integrated Software Modules (PRISM)	It was an architecture method that can be effective for the COTS selection.	(Lichota et al., 1997)
4	Comparative Evaluation Process (CEP)	It emphasized on the reliability of data sources by using the confidence factor (CF). The high value of CF indicates high reliability of the data source.	(Phillips & Polen, 2002)
5	COTS-based Requirements Engineering (CRE)	It addressed the importance of using the non-functional requirements in the COTS software evaluation process.	(Alves et al., 2000)
6	COTS software Acquisition Process (CAP)	It was based on hundreds of quality metrics to customize or tailor the evaluation process and relied on the team's available effort and experiences.	(Ochs et al., 2001)
7	COTS-Aware Requirements Engineering (CARE)	It was to evaluate and select the COTS products in an objective manner.	(Patricia et al., 2001)
8	Requirements-driven COTS Product Evaluation Process (RCPEP)	It focused on the requirements engineering phase, which was based on the variety of agents' view to elicit flexible requirements.	(Chung et al., 2001)
9	Plan, Establish, Collect, and Analyze (PECA)	It was from SEI to customize the COTS products selection process by defining a high level processes and providing guidelines to help in the tailoring process.	(Comello-dorda et al., 2002)
10	Balanced Reuse Model (BAREMO)	It used the AHP technique to rank and select the COTS software products. (Lozano-Tello & Gomez-Perez, 2002)	(Lozano-Tello & Gomez-Perez, 2002)

11	The Storyboard method	It utilized the 'use case' and 'screen-captures' in eliciting user's requirements that helps users to understand their demands and select more suitable COTS software.	(Gregor et al., 2002)
12	Combined Selection of COTS Components approach (CSCC)	It attempted to choose from various available COTS software to achieve user's requirements.	(Burgues et al., 2002)
13	The WinWin Spiral model	It focused on the iterative process to determine, analyze, and solve any encountered risk.	(Boehm et al., 2003)
14	The fuzzy theory	It suggested to transform qualitative data into quantitative data by using suitable technique to determine optimal solution.	(Erol et al., 2003)
15	The DesCOTS system	It relied on the quality models such as ISO/IEC 9126 to evaluate COTS software through the integration of various tools, was created.	(Grau et al., 2004)
16	The Compatible COTS Component Selection (CCCS) model	It emphasized on the integration and fitness of the COTS software.	(Bhuta & Boehm, 2005)
17	COTS Software Selection Process (CSSP)	It was specially put forward in 2006 for the US Department of Energy to evaluate and rank the COTS software through systematic steps in determining the best solution with manageable risk, besides enabling a customization process in meeting the requirement of different projects.	(Lin et al., 2007)
18	The Mismatch-Handling aware COTS Selection (MiHOS)	It was developed based on the general COTS Selection (GCS) process. This approach focused on handling the mismatches between user's requirements and the COTS software through linear programming adaptation.	(Mohamed et al., 2008)
19	Uncertainty Handling in COTS Selection (UnHOS)	It was introduced to address the uncertainty issues and how they affect the COTS quality and user's satisfaction.	(Ibrahim et al., 2011)

Next section explains the most related theories and techniques that have been used in the framework construction.

2.3.2 COTS Software Evaluation and Selection Theories

A diversity of theories were used and adapted to develop the reliable and rigor method of the COTS software evaluation and selection. These theories are presented in the following sections.

2.3.2.1 Evaluation Theory

There are several theories and methods have been proposed to address the evaluation issue in different disciplines such as theory-oriented evaluation, social science theory, and program theory, which resulted in not developing a general theory that

provides all required evaluation components (Lopez, 2000; Scriven, 1991; Zarour, 2009). In addition, some of them have been used successfully in some fields while they failed in others.

Therefore, according to Zarour (2009), the evaluation methods in the software engineering (SE) field were developed and performed without considering the efforts and lessons learned from other fields. Lopez (2003) pointed out that by studying theories and methods of evaluation that have been developed in other disciplines helps to develop more systematic and completed evaluation methods that can apply in various areas of SE. Thus, the evaluation theory was successfully used as a general theory in many fields and disciplines for several reasons which are:

i. It has been developed based on the knowledge provided by the previous theories that have been successfully used in different areas and disciplines.

ii. It provides the basic six general components involved in any evaluation process, which are: evaluation target, evaluation criteria, yardstick, data collection techniques, synthesise techniques, and evaluation process.

iii. It encompasses diverse of methods, techniques, and approaches for collecting and synthesizing the data in different fields.

Basically, the evaluation theory emphasizes that the development of an evaluation method is based on the six fundamental components: evaluation target, evaluation criteria, yardstick or standard, data gathering or assessment techniques, synthesis

techniques, and evaluation process (Figure 2.5). The definition and description of each component is as follows (ACUÑA, 2001; Lopez, 2000; Scriven, 1991; Zarour, 2009):

1. **Evaluation Target**: Defining and delimitating the target that represents the objectives of the study under evaluation.

2. **Evaluation Criteria**: Determining the evaluation criteria is essential and critical in analyzing the characteristics of the evaluated target. The selected criteria are then structured into a diagrammatic tree known as criteria tree. This criteria tree is used to develop the yardstick.

3. **Yardstick or Standard**: The inference from the criteria tree creates a yardstick for the evaluation process. At this stage, the target is at its ideal condition in which the specifications (values) of all defined criteria are incorporated. The representation of these specifications is in the form of a pair structure comprising of a criterion and datum/information. To reach positive evaluation, the yardstick must include the threshold values to indicate the minimum value for each criterion.

4. **Data Gathering/Assessment Techniques**: The techniques are required to collect data or information for each evaluated criterion. The evaluation criteria and yardstick is used as a basis to select the most appropriate data collection technique. The technique can be more than one depending on the criterion value type specified in the yardstick.

5. **Synthesis Techniques**: A set of techniques needed to organize and synthesize the information obtained from the gathering techniques.

6. **Evaluation Process**: A series of activities and tasks to perform the evaluation process that comprises of the following phases:

    - Planning: This phase covers the target to be evaluated, evaluation delimitation, and execution planning management.
    - Examination: The data gathering techniques are applied in this phase to obtain the data for examining the target to be evaluated.
    - Decision making: In this phase, the synthesis technique is executed to get the final result and prepare the final report.

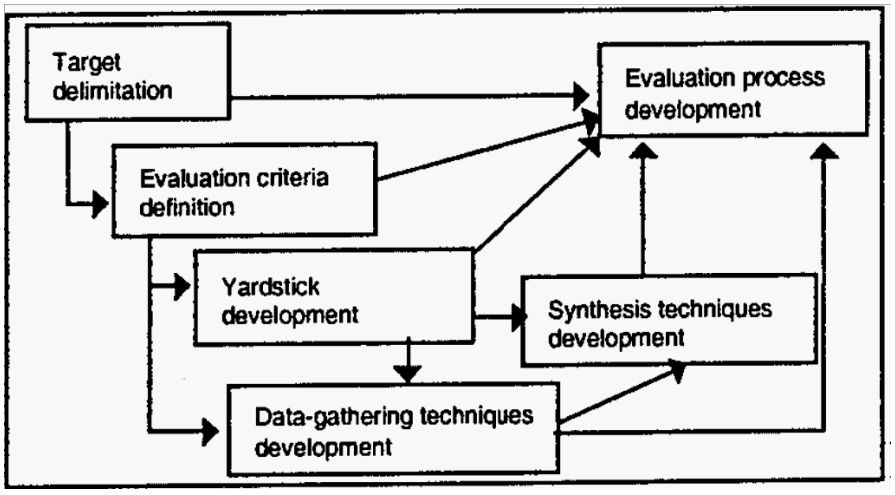


Figure 2.5. Evaluation Theory Components

(Scriven, 1991)

One of the successful used of the evaluation theory in software engineering is the works that were carried out by Ares et al. (2000) and Zarour (2009). They used the

evaluation theory to develop a framework for evaluating the methods of Software Process Assessment (SPA). Moreover, Casal et al. (1998) used the evaluation theory to develop a formal and systematic framework for evaluating the software process of an organization. Therefore, the evaluation theory was used in this research by adapting the six basic evaluation components to construct the appropriate framework for COTS software evaluation and selection.

2.3.2.2 Multi Criteria Decision Making (MCDM)

Multiple criteria are usually used when selecting the best fitness among various alternatives whereby some criteria are more important compared to the others. Since the decision-making process of the COTS software evaluation and selection depends on evaluating a number of alternatives, MCDM offers the most appropriate technique for decision makers (Alanbay, 2005; Gupta et al., 2012; Javed et al., 2012; Jie et al., 2005; Ncube & Dean, 2002).

In this context, MCDM is the most well-known branch of decision making. It was introduced as a promising and an important field of study in the early 1970s to deal with complex engineering problems under the presence of multiple decision criteria. In the real life, there is no ideal alternative considered as the optimal for each objective. Therefore, the main aim of MCDM is to find the best compromise. MCDM is concern with breaking a problem down into its constituent parts or components and establishes preferences for evaluating, prioritizing to rank available alternatives and select the most suitable alternative in terms of multiple criteria (Gomes et al., 2008). MCDM consists of the following components:

- Alternatives: These represent the available choices of actions to decision makers. These alternatives should be limited (from several to hundreds), screened, prioritized and eventually ranked.
- Multi criteria: For each problem, there is a set of criteria, attributes, or goals. They represent the several dimensions of alternatives' views. For example, purchasing a new house could be viewed based on the location, price, size, and others.
- Incommensurable units: Different units of measure associated with different criteria.
- Criteria weights: Since the criteria do not have the same importance, they need to be distinguished through assigning weightage that represent their importance. Typically, these ordinal or cardinal scale weightage are normalized to add up to one.
- Ranking the software alternatives based on how they will fit or satisfy the evaluation criteria.

The most common MCDM techniques that have been used in COTS selection are weighted sum or scoring method (WSM), Analytical Hierarchy Process (AHP), outranking method, and fuzzy multi criteria decision making (Kunda, 2002; Ruth, 2008). These techniques are explained and compared in Table 2.3.

Table 2.3

The MCDM Techniques

MCDM Techniques	Main Idea	Strength	Weaknesses	References
WSM	It calculates the overall score for each COTS software alternative against the evaluation criteria by summing up the criteria weights multiplied by their respective value when selecting software.	• Very simple • Suitable for small problems	• Lack of process to assign the weight • Does not support the large number of evaluation criteria • Weak in sustaining multi-value features • Include the summation of different data types (i.e. cost plus memory plus quality)	Alves & Finkelstein, 2003; Maxville et al., 2004; Solberg & Dahl, 2001
Outranking	It ranks each alternative on every attribute and determines an outranking relationship to classify the attributes into preferred and non-preferred. Once the comparison of each attribute is made, the units and attributes are eliminated.	• Suitable for uncertainty problem	• Some issues with explicating the causing of decision and complete ranking may not be achievable • Only indicates the preferred alternative not the preference value	Roy, 1991; Buchanan & Vanderpooten, 2007; Kunda, 2003; Ruth, 2008
Fuzzy multi-criteria	It handles and quantifies the vagueness of users' thought and perception, where the linguistic terms are represented by the approximate reasons. It uses the linguistic terms to assign the weights of the criteria and the scores of the software alternatives	• Suitable for subjective problem	• Associated with criteria whose values are not numbers, but words or sentence in natural language • Does not support the objective data	Lin et al., 2007; Ruth, 2008
AHP	It decomposes the evaluation criteria and estimating the software alternatives using the hierarchy structure where the weights and the score of the alternatives are calculated through the pairwise comparisons.	• Suitable for simple and complex problem • Support the objective and subjective data • Ranking the alternatives based on their fitness to evaluation criteria • Provide high efficient method to assign the weights	• Does not deal with mismatches problem • Alternatives re-estimation is required once new alternatives are found.	Saaty, 2008; Falessi et al., 2011; Sun, 2010; Ruth, 2008; Jadhav & Sonar, 2008

Analytical Hierarchy Process Technique (AHP)

Description

Thomas Saaty developed the AHP technique in 1980 for multi-criteria decision-making. The main advantage of AHP is that it enables the decision makers to arrange the evaluation criteria into a hierarchy. In the hierarchical structure, the main goal is located at the top, the criteria and sub-criteria are at the middle, while the competing alternatives are at the bottom (see Figure 2.6) (Pogarcic et al., 2008). The pair-wise comparisons are used to estimate the relative importance of those criteria at each level of the hierarchy. The comparison results are converted into normalized rankings that represent the weight of compared criteria. Similarly, the final score of all COTS alternatives can be estimated using the pair-wise comparisons with respect to each criterion (Mujeeb-U-Rehman et al., 2005; Ngai & Chan, 2005; Saaty, 2008).

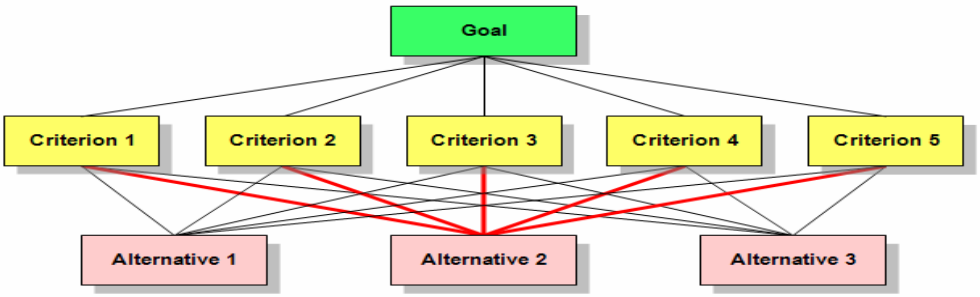


Figure 2.6. The AHP Hierarchical Structure

(Pogarcic et al., 2008)

Advantages

The AHP technique has many advantages over the other methods, which made it the most common MCDM method used in the decision making process through several fields. These are depicted in Table 2.3. Since the AHP has become the most common

technique used in several related studies, this research has opted to the technique due to the following advantages. Firstly, the technique uses human judgments rather than depending on the underlying information when performing the evaluation, which allows users to participate in making decision. Secondly, it converts these evaluations into numerical values that can be processed and compared over the entire problem. The AHP technique also provides a comprehensive and rational framework for structuring a problem, representing and quantifying its elements, relating those elements to the overall goals, and evaluating the alternative solutions (Cangussu et al., 2006; Dolan, 2010; Falessi et al., 2011).

The main strength point of the AHP is that it relies on the pairwise comparison to determine the weights of the evaluation criteria. This will certainly reduce the burden on the experts and provides consistent analysis of the comparisons and weights. The consistency checking mechanism in the AHP technique increases the reliability and quality of the information on the criteria and alternative to be estimated as well as reducing possible judgment errors. The weight value is derived from each criterion in the hierarchy, which provides diverse and often make the incommensurable criteria to be compared to one another in a rational and consistent way. Finally, the AHP technique is considered as an effective ranking approach where the alternatives are ranked based on their fitness to the evaluation criteria (Dolan, 2010; Eldrandaly, 2007; George et al., 2008; Ibrahim et al., 2011; Jadhav & Sonar, 2009). Based on these advantages, the AHP technique has successfully been applied in several fields to make appropriate decision. Table 2.4 shows some examples of the AHP applications in different fields.

Table 2.4

The AHP Applications in Different Fields

The Area of Application	Decision Context	Citation
Prioritization of sites/areas for industrial/military activity	Land condition assessment for allocation of military training areas	Mendoza et al. (2002)
	Selection of boundaries for national park	Sharifi et al. (2003)
Environmental/remedial technology selection	Regulation of water flow in a lake–river system	Hamalainen et al. (2001)
Natural resource management	Management of small forest in North Carolina, USA	Rauscher et al. (2000)
Software selection	ERP system	Bharathi et al. (2012); Ünal and Güner (2009); Wei et al. (2005)
	Select the most appropriate tool to support knowledge management (KM)	Ngai and Chan (2005)
	Evaluating and selecting a design and manufacturing software package at a Canadian cabinet manufacturing company	Assadi and Sowlati (2009)
	Selecting a third party logistics (3PL) provider	Daim, et al. (2013)
Education	Evaluate curriculum in Department of Risk Management and Insurance at Universities	Fan and Cheng (2009)
Industry	Evaluating the manufacturing industry competitiveness	Singh et al. (2011)
Business management	Selection criteria of recruitment for information systems employees	Hsiao et al. (2011)

Many existing methods for COTS software selection, such as STACE, PORE, and CRE, have widely adopted the AHP technique to synthesize the data and rank the relevant alternatives (Land et al., 2008; Ibrahim et al., 2011).

Issues associated with AHP

The AHP technique is not the silver-bullet or ideal technique in decision making. Although the AHP technique is the widely used in the COT software evaluation and decision-making, there are several associated issues and limitations raised by previous studies.

- According to Akhavei and Hayes (2003); and Becker and Rauber (2009), the AHP technique involves a time and effort consuming because it requires a number of pairwise comparisons based on the number of the evaluation criteria and alternatives. The increased the number of criteria or alternatives increases more time and effort. This issue will reflect negatively on the sources constraints especially the available time for the project.

- Despite offering the best mechanism to assign criteria weightage and the checking on the consistency of user's judgments in order to provide results that are more reliable, the AHP technique is incapable of assigning accurate score for each criterion of the software alternative. This drawback will influence the real degree of alternative achievement from this criterion. In this context, Tsai et al. (2010) and Sun (2010) have pointed out that the AHP technique failed to deal with the gaps or mismatches between user's requirement and the software components, which eventually resulted in inaccurate decision.
- In the context of software components especially in the COTS software, the market is always evolving where new software products with new capabilities are produced rapidly. This increases the possibility of changing (add or delete) the number of the current software alternatives evaluated by the AHP technique. In this case, the pairwise comparisons for all alternatives need to be redone because the AHP technique calculates the score based on the pairwise comparison between those alternatives at each criterion.

Eventually, the final alternatives ranking will also change (Ruth, 2008; Jadhav & Sonar, 2008).

Based on these issues, this technique has always been questioned in terms of its usefulness in the COTS software evaluation and selection particularly when the time and experience are limited and the accuracy of the final decision is required. In this case, the specialty of the AHP technique is represented by its ability and flexibility to be combined with different techniques in extracting advantages of the desired objectives in a better way (Felice & Petrillo, 2012; Marianos et al., 2011; Vaidya & Kumar, 2006). Therefore, the AHP technique is adapted in this research to integrate with other suitable technique so that a more useful technique can be developed that can handle and solve the COTS software mismatches issue.

2.3.2.3 COTS Mismatches

The mismatches between the COTS software features and user's expectations often arise because most of the COTS software are developed for a more general usage, while user's requirements are based on specific characteristics or needs (Alves & Finkelstein, 2003; Carney et al., 2000). The COTS mismatches are usually classified into four types as suggested by (Alves et al., 2005; Kiv et al., 2010):

- Differ (partially mismatching) – Differ mismatching means that there are partial matching between user's requirements and the COTS software. For example, if the user's requirement is stated as "the COTS software shall save

file in DOC and PDF format file"; the COTS software on the other hand does not support the PDF format.

- Fail matching (full mismatching) – Fail matching refers to a situation whereby the COTS software completely fails to achieve user's requirement. For example, the user's requirement statement goes like this: "the COTS software shall save the modifications automatically" but the COTS software does not support such requirement.
- Fulfil matching (zero mismatching) – Zero mismatching denotes that the COTS software completely achieved user's requirements. For example, if the user required that "the new system should support protection against external attacks", the COTS software can fully satisfy the requirement by providing an anti-virus scanning mechanism.
- Extend - This type of mismatch occurred when the COTS software provide some extra services or functionalities that are not requested by user. In this context, the extra features of the COTS software will give either helpful, hurtful, or neutral impact on the system.

In this research, all of the previous classifications of the COTS mismatches were taken into consideration and appropriate mechanism was proposed in chapter 5. This is necessary because only identifying the type of mismatches is not enough to determine the most appropriate COTS software. Therefore, it is critical to determine the mismatch solution as well as the constraints of the required resources such as the costs and associated risks when applying these resolutions (Alves, et al., 2005;

Mohamd et al., 2008). In this context, two scenarios can be used to solve the COTS mismatches:

1. Requirements modification: This is an attempt to customize the mismatched between user's requirements and the COTS software features. To do that, the requirements should be more flexible and less specific, and determining the requirements is based on the available COTS software features in the market (Alves & Finkelstein, 2003; Kiv et al., 2010; Ncube & Dean, 2002). In fact, it is inevitable to achieve the best balanced between the requirements flexibility and accuracy.

2. The COTS software adaptation: In this case, the COTS software should be customized to meet user's requirements by using several resolution actions as stated by (Kiv et al., 2010; Mohamd et al., 2008):

    - Add-ons: this action aims to use additional add-ons to provide functionality to the COTS software.
    - Application Programming Interface (API): The API strategy enables the development of a controlled program that can perform the COTS functions.
    - Scripting language: The identified mismatches may be solved by using a scripting language that is supported by the COTS software in writing the custom code.

- GUI-based: The GUI interface is used in tailoring the COTS software according to user's desired attributes.
- Parameters-based: The tailoring of the COTS software is based on its parameters adjustment in satisfying user's requirements.

As mentioned early, although MCDM methods (e.g. AHP and WSM techniques) are the most popular techniques that used by the majority of existing COTS selection methods to consolidate the data and rank the COTS alternatives based on their fitting to the user's requirements (Eldrandaly, 2007; Land et al., 2008; Pande, 2012). These methods have a lack to deal with the mismatches occurred between COTS capabilities and user's requirements because of underlying assumptions and their judgment value systems (Ncube & Dean, 2002; Sun, 2010; Tsai et al., 2010).

Although they are the most popular techniques used in majority of the existing COTS selection methods (Eldrandaly, 2007; Land et al., 2008; Pande, 2012), the MCDM methods (e.g. AHP and WSM techniques) is still have a lack especially in handling the mismatches between the COTS capabilities and user's requirements. This setback occurs because of the underlying assumptions and the judgment value of the systems (Ncube & Dean, 2002; Sun, 2010; Tsai et al., 2010). Therefore, to overcome such related problem, the following techniques, which are considered as the alternative of MCDM, are applied to handle the mismatches problem.

1. Fulfillment Cost

The fulfillment cost technique aims to establish the cost of fulfilling the gaps identified between the COTS software and user's requirements. Based on the parameters and identified gap, there are several strategies can be generated and followed (Comella-Dorda et al., 2004; Ncube & Dean, 2002), as shown in Figure 2.7.

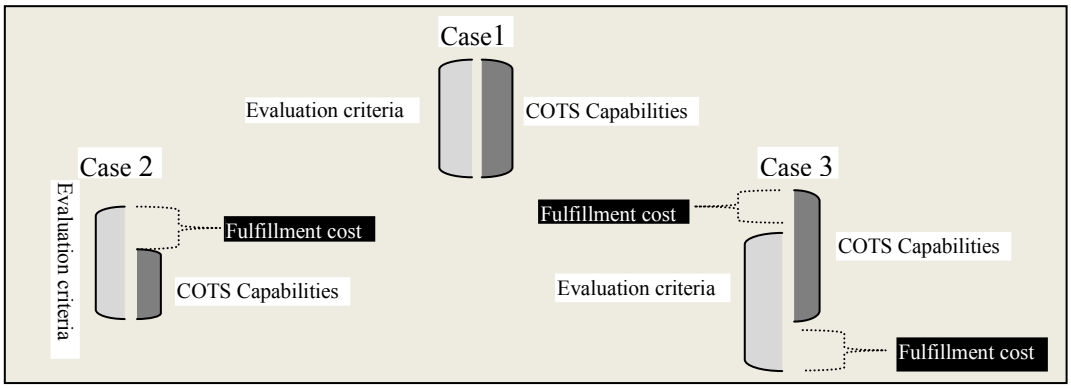


Figure 2.7. The Fulfilment Cost

(Ncube & Dean, 2002)

In Figure 2.7, case 1 presents an ideal case of the COTS capabilities that fully matched user's requirements. This indicates that the gap and cost of fulfillment does not exist. In case 2, the COTS software partially matches user's requirements and the cost of fulfillment is estimated to meet the remaining sections of the requirements. Case 3 signifies that the COTS software does not only fail to match user's requirements but also provides extra functionalities that exceeded the requirements boundaries. In this case, two kinds of costs are identified: the fulfillment cost of meeting user's requirements; and the fulfillment cost of handling (control, manage, hide, etc.) the extra functionalities.

The cost of fulfilling the gap has several facets such as wrapper, custom code, training, processes changing, documentation, and hidden extra capability of COTS (Comella-Dorda et al., 2004; Ncube & Dean, 2002; Wallnau et al., 2002). This type of cost fulfillment approach focuses only on estimating the strategies cost of dealing with identified gaps. The approach, however, does not include the important role of other factors in determining the fitness of the COTS software such as the risks, times and efforts of implementing the fulfillment strategies.

2. Gap Analysis (GA)

The business dictionary defines GA as "*a methodical tabulation of all the known requirements of consumers in a particular category of products, together with a cross-listing of all the features provided by existing products to satisfy these requirements*" (Ncube & Dean, 2002). J. Michael Scott, at the University of Idaho, generated the basic idea of the GA process in 1980s. He firstly developed this method to assess the endangered birds in Hawaii (Bordley, 2001).

In general, GA refers to the activities of studying or identifying the differences or space between the current state and the ideal or target state. These differences or spaces are called gaps and these gaps should be filled or bridged to reach the target state (see Figure 2.8) (Berch, 2003; Bordley, 2001). The gap may take several forms or types such as awareness gaps, knowledge gapes, implementation gaps, and commitment gaps.

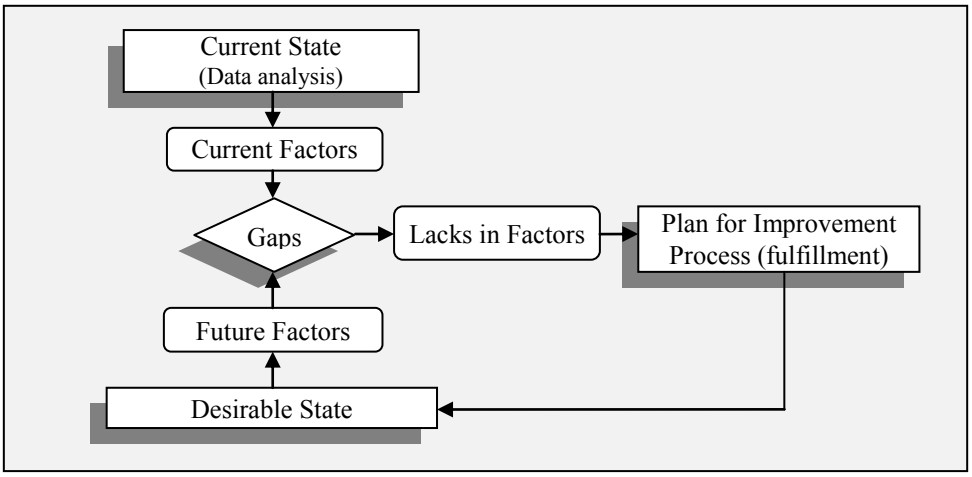


Figure 2.8. General Idea of GA

(Adapted from Bordley, 2001)

Typically, the COTS software is developed to satisfy the general requirements of the users instead of fulfilling individual requirements (Alves & Finkelstein, 2003; Brownsword et al., 2000). Therefore, there is need to specify a method to identify and analyze the differences (gaps) between user's requirements (expectations) and the products offered by COTS. In this context, GA provides sense of "best" and helps to identify significant issues in trying to meet the corresponding requirement (Comella-Dorda et al., 2004).

The GA requires deep analysis to study both the current state which is represented by the features of the COTS software and desirable or target state represented by user's expectations. The analysis is to identify the gaps as well as their fulfilment solutions (Chung et al., 2001). To apply the gap analysis technique, user's requirements need to be assessed against the COTS features in order to identify those capabilities that are not fulfilled (Ncube & Dean, 2002; Sheng & Wang, 2008). This is done through the use of the matrix of two dimensions. The COTS products are sorted in the first

row of the matrix versus the user's requirements in the first column. The matrix cells demonstrate the fitness of each COTS candidate against each criterion by filling the cell with text that described how well a COTS product provide the function (Comella-Dorda et al., 2004; Sheng & Wang, 2008).

Advantages

According to Comella-Dorda et al. (2004) and Sheng and Wang (2008), the main benefit of the GA technique if the assistance offered to evaluators in identifying and determining the important criteria; understanding the services or functions provided by the COTS candidates; and estimating the differences. By putting the COTS candidates side by side during the sorting of comparison can facilitate team developers to reason out related strengths and weaknesses. It can also clarify the overall patterns and give some sense of global superiority.

The comparison of the fulfilment cost using the GA technique provides general understanding about the existed gaps. The measurement can be performed through the use of a simple "Yes or No" matrix. The measurement matrix enables evaluators to be aware of the consequences of those identified gaps (Chung et al., 2001, Comella-Dorda et al., 2004). Figure 2.9 presents the example of matrix that contains some of the fitness measurement.

Products / Criteria	P1	P2	P3
C1	0.30	1.00	0.95
C2	1.00	1.00	1.00
C3	0.80	0.60	1.00
C4	1.00	1.00	0.10
C5	0.03	1.00	0.45

Figure 2.9. Fitness Measurement Matrix

The GA technique also provides additional information regarding the actual level of fulfilment process as shown in Figure 2.10 (Chung et al., 2001, Comella-Dorda et al., 2004).

Products / Criteria	P1	P2	P3
C1	Inaccurate math	Complete solution	Precision only to 2 decimals
C2	Complete solution	Complete solution	Complete solution
C3	Vendor out of country	Vendor Canadian	Complete solution
C4	Complete solution	Complete solution	Limited Java support
C5	Linux Platform Required	Complete solution	Windows Only

Figure 2.10. Matrix of Actual Information about Gap Fulfilment

Based on the above descriptions, the GA is proven to be most useful technique for identifying, analysing, and determining the required costs and efforts in fulfilling the gaps between the COTS candidates and user's criteria to support suitable COTS product selection. The GA technique offers other kinds of information required in the gap fulfilment, which include, existing gap (Yes/No), level of gap, and fulfilment solution and costs (Comella-Dorda et al., 2004; Ncube & Dean, 2002; Sheng & Wang, 2008).

Although it was used widely in several fields such as education (Quality of college students' expectations) (Jackson et al., 2011), business service (Nguyen et al., 2009), enterprise architecture (Postina et al., 2009), information quality assessment (Lee et al., 2002), and service quality (Chen McCain et al., 2005), the GA technique had not been widely applied in the field of COTS software evaluation. Since the GA technique was developed to address the gaps or mismatches between the ideal state (user's expectations) and current state (software offered), this support the idea that

the gaps or mismatches problem is still insufficiently handled in that field (Land et al., 2008; Chung & Cooper, 2004; Ncube & Dean, 2002).

Issues associated with GA

According to Comella-Dorda et al. (2004); and Ncube and Dean (2002), GA technique has a lack to provide how to compute the total fitness of the software against the evaluation criteria specially when there are many criteria with different data type measurements. Moreover, this technique only deals with low level of evaluation criteria and assumes all of them the same important. Furthermore, the results of this technique can be sometime very vague.

Since the GA technique has been accepted as an appropriate technique to identify mismatches, it still needs to be supported by a good method in prioritizing the evaluation criteria. This is necessary to distinguish between the important and unimportant criteria. It must aggregate all the individual matching level against each criterion. The technique also needs to consider the constraints of identified mismatch resolutions such as costs, time, and risks when determining the matching level between the criteria and COTS capabilities. Considering such capabilities, the GA and AHP technique are adapted in this research to overcome the previous limitations and provide a more reliable and effective technique for handling COTS mismatches.

2.3.2.4 The General COTS Selection (GCS) Process

As mentioned in the previous studies, there are several processes, activities and techniques that are shared by the existing methods for the COTS software evaluation

and selection purposes. These processes and activities can be executed either iteratively, sequentially, or overlapping (Land & Blankers, 2007). According to the analysis on the existing methods conducted by (Fahmi & Choi, 2009; Jadhav & Sonar, 2008; Land et al., 2008; Land & Blankers, 2007; Mohamed et al., 2007), the common processes and activities of COTS software evaluation and selection can be described in terms of three main processes: preparation; evaluation; and selection process (Figure 2.11).

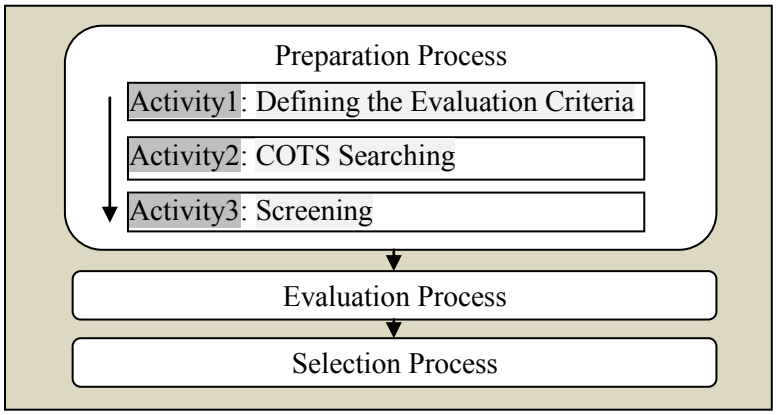


Figure 2.11. The COTS Selection Process

(Adapted from Javed et al., 2012)

1. Preparation Process

In this process, each evaluation criteria and potential COTS candidates is identified for further detail evaluation (Land & Blankers, 2007). All of these are done among the following activities:

- **Activity 1:** Defining the Evaluation Criteria

Defining the evaluation criteria is a prior activity before starting the actual COTS software evaluation. This activity aims to decompose the organizations'

requirements by analyzing different requirements' sources against the COTS software features. The sources include the application requirements, application architecture, project objectives and constraints, organization infrastructure, and software market. The analysis is performed to determine the hierarchical structure starting from the high-level criteria in the top until the low-level criteria at the bottom (Jadhav & Sonar, 2009; Kaur & Mann, 2010; Clough, 2003).

- **Activity 2:** COTS Searching

Searching activity is an attempt to identify and determine all possible alternatives of the COTS software based on the initial criteria known as the search criteria. These criteria are typically based on the required functionality and key constraints (Mohamed et al., 2007). The search criteria must be broad and distinguished in order to collect various COTS software with different features, and to ensure that the searching is not limited by too many constraints (Cechich & Piattini, 2007; Cechich & Taryano, 2003). Moreover, several sources of search should be used to elicit the COTS software. The sources can be either internal such as in-house components libraries, or market surveys such as internet, magazine and journal vendors, trade show and conferences, publication and sales promotions (Kunda & Brooks, 1999; Yanes et al., 2012).

- **Activity 3:** Screening

Typically, the output of the search activity is often too general and consists of a very large list of COTS alternatives, which makes the estimating and analyzing of all appropriate characteristics of any alternative takes more time and effort as well as

increases the costs of the evaluation process (Cechich & Piattini, 2007; Kunda & Brooks, 1999; Phillips & Polen, 2002). Therefore, it is necessary to select the most promising and cost–effective COTS software alternatives in order to minimize the searching list so that detailed estimation can be performed using available resources (e.g. budget and time). This eventually decreases the overall duration of the evaluation and selection process (Javed et al., 2012; Kontio, 1996; Mohamed et al., 2007).

2. Evaluation Process

The evaluation process represents the core of the COTS evaluation and selection process. This process aims to estimate each COTS software alternative against the evaluation criteria in more detail. These alternatives are then sorted based on their importance in order to assist the decision makers in selecting the fitness of the COTS software (Kontio, 1996; Mohamed et al., 2007).

Typically, the quality of the evaluation process is also depending on a good data collection technique. A variety of criteria and situations require multiple data collection techniques from various sources (Hill et al., 2004; Phillips & Polen, 2002). Among the techniques used for the data collection purposes are documents review, market survey, experiments, and pilot study (Hill et al., 2004; Kunda, 2002; Phillips & Polen, 2002). The document review is used to evaluate the COTS software based on vendor information that can be found from technical documents, brochures, and websites. The market survey is conducted through questionnaires or interviews with vendors and users community.

3. Selection Process

As mentioned earlier, the outputs of the COTS software evaluation process can be produces in several forms such as facts, checklists, weights, opinions, and others. These forms of output should be consolidated and interpreted into information by using suitable technique in order to facilitate analysis and decision making (Hill et al., 2004). Data analysis is a creative task that depends on the evaluators skills in obtaining relevant data from the evaluation process (Hill et al., 2004; Mohamed et al., 2007).

Even though the previous processes and activities are the common and shared between most of the existing methods for COTS evaluation and selection, there are many processes are required to provide a well-defined evaluation process such as the planning process. In the planning process, several activities related to the project target, team, and project constraints are needed to be clarified in systematic way. In addition, the screening activity in the preparation process and the data collection activity in the evaluation process consume most of project time, while they can be improved by integrating them to gather in one activity to save the time and effort during the evaluation process. Therefore, the previous processes and activities need to improve in order to provide a well-defined and systematic process for evaluating and selecting COTS software.

After studying the main theories, techniques, and main processes and activities in the COTS software evaluation and selection process, it's necessary to identify how the

previous methods for COTS software evaluation and selection addressed these elements, which are explained in the next section.

2.3.3 The Missing Elements in the Existing Methods for COTS Evaluation and Selection

It is can be derived from the previous frameworks that all methods are sharing similar iterative and overlapping steps which include evaluation criteria definition, alternatives searching; alternatives screening; alternatives evaluation; data evaluation synthesis and fitness selection (Javed et al., 2012).

However, despite the above similarities, those methods inadequately addressed many issues such as identifying the mismatches between user's requirements and the COTS software features, and providing proper evaluation criteria based on the non-functional requirements (Mohamed et al., 2008). Table 2.5 shows the comparison between these methods based on several factors such as addressing, defining evaluation target, defining yards tide providing well defined evaluation processes, defining the data gathering technique, defining the synthesis technique, and using software tool support.

Table 2.5

The Comparison of Existing Methods for COTS Selection

Methods		COMPARISION FACTORS						
Year	Name	Evaluation Target	Defining Yardstick	Proposing Evaluation Processes	Proposing Evaluation Criteria	Using Data Gathering Technique	Using Synthesis Technique *(Based on Addressing COTS Mismatches)*	Support Tool
1995	OTSO	Off-The-Shelf Software (OTS) components	Nil	-Define evaluation criteria -Search -Screening -Evaluation -Analysis of results	Relies on the ISO 9126 and cost criteria	Nil	Relying on AHP technique *(doesn't addressing COTS mismatches (refer to section 2.3.2.2))*	Nil
1997	IusWare	COTS software	Nil	- Problem formulation - Design evaluation model - Apply evaluation model	Relies on the ISO 9126 quality model and ignore other criteria	Nil	Outranking aggregation technique *(Lack of addressing COTS mismatches)*	Nil
	CISD	Not addressed	Nil	-Identification -Evaluation -Integration	Focuses on functional requirements and interoperability criteria	Nil	Nil	Nil
	PRISM	COTS & GOTS products	Nil	-Identification -Screening -Stand-alone test -Integration test -Field test	Suitability criteria (generic architecture, maturity, maintainability, cost, portability, scalability)	Paper analysis	Nil	Nil
1998	PORE	COTS software	Nil	-Requirements acquisition -Supplier selection -Software selection -Contract production -Package acceptance	-Architecture requirements -Product supplier (e.g. technical capabilities, experience, standard certification) -Legal issues (e.g. contract terms, licensing arrangements)	Card sorting and laddering	Defined set of techniques such as Multi-criteria decision making techniques. *(don't addressing COTS mismatches (refer to section 2.3.2.2))*	PORE Process Advisor tool

1999	STACE	COTS software	Nil	-Requirements elicitation -Social-technical criteria defining -Alternatives identification -Evaluation	-Technology factors -Business issues -Customer capability -Marketplace variables -Vendor capabilities (the details in table 2.6)	Documents review and brainstorm meeting	Relying on AHP technique. *(doesn't addressing COTS mismatches (refer to section 2.3.2.2))*	Nil
	CEP	COTS software	Nil	-Scoping evaluation effort -Searching/screening -Criteria definition -Evaluation & analysis	Functional, performance, architectural, financial and management criteria.	Nil	Simple weighted-average theory. *(doesn't addressing COTS mismatches)*	Nil
	CRE	COTS software	Nil	-Identification -Description -Evaluation -Acceptance	-Quality criteria -Non-technical criteria such as cost & benefits, capabilities, risk analysis, vendor reputation, vendor infrastructure, and vendor support	Nil	Using WSM and AHP techniques *(don't addressing COTS mismatches (refer to section 2.3.2.2))*	Nil
2000	CAP	COTS software	Nil	-Initialization -Execution -Reuse	-Functional (suitability, accuracy, interoperability, and security) -Non-functional (reliability, usability, efficiency, portability) -Domain/architecture (domain compatibility & architecture compatibility) -Strategic (cost & risk)	Goal Question Metrics (GQM); expert knowledge elicitation ; and subjective measurement	Relying on AHP technique *(doesn't addressing COTS mismatches)*	Nil
2001	RCPEP	COTS software	Nil	-Trade study -Hands-on evaluation	-Functional requirements	Questionnaire	Weighted averages technique *(doesn't addressing COTS mismatches (refer to section 2.3.2.2))*	Nil
	CARE	COTS software	Nil	-Define goals -Match goals -Rank components -Negotiate changes -Select component	-Functional requirements -Domain -Vendor -Standards compliance -Performance -Security	Nil	Gap analysis technique *(It doesn't show how the mismatches influence on the decision making)*	CARE Assistant Tool

2002	PECA	COTS software	Nil	-Planning -Establishing criteria -Collecting data -Analyzing results	-Product characteristics -Vendor characteristics -Hardware Configuration -Standards -Software Configuration -Functionality -Licenses	Literature review, vendor appraisal, and hands-on	Delphi technique and AHP technique *(don't addressing COTS mismatches)*	Nil
	BAREMO	Software component	Define the threshold values.	Following AHP steps.	-Product time -Cost rating -Final product quality -Development risk	Nil	Relying on AHP technique *(doesn't addressing COTS mismatches)*	BAREMO Tool
	StoryBoard	COTS software	Nil	Develop storyboard using screen captures and use-cases (no formal evaluation process)	-Functional requirements *Non-functional requirements are not addressed*	Use-cases and screen captures	Nil	Nil
	CSCC	COTS software	Nil	-Plan the selection process -Identify COTS candidates -Identify global COTS integration scenarios -Evaluate individual scenarios -Evaluate integration scenarios (global level). -Select the COTS products	Not well-defined	Nil	Based on the method used at the local level	Nil
2003	Win Win spiral	COTS software	Nil	-Identification (stakeholders, objectives, and constraints) -Evaluation -Elaboration -Verification & validation -Stakeholders' review	Non-functional requirements are not addressed	Nil	AHP (*Addresses the risks associated with COTS mismatches*)	Nil

	The Fuzzy Theory Approach	COTS software	Nil	-Acquire requirements and product information -Transform qualitative data to quantitative data -Construct a multi-objective model -Solve the model	Not well-defined	Nil	Fuzzy QFD (Quality Function Deployment) *(focused on the linguistic terms and has a lack of handling COTS mismatches)*	Nil
2004	DesCOTS system	COTS software	Nil	Support GCS	Focused on quality criteria supported by ISO/ICE 9126	Nil	Using AHP techniques *(doesn't addressing COTS mismatches)*	COTS Selection Tool
2005	CCCS	COTS software	Nil	Support GCS	Non-functional requirements are not addressed	Prototyping	Nil	Nil
2006	CSSP	COTS software	Nil	-Form evaluation team -Apply team non-software process -Identify COTS criteria -Apply level I filter using published vendor information -Apply level II filter based on vendor demonstration -Analyze data	-Functional requirements -Vendor qualification (reputation, financial status, and competitors) -Product quality -Interoperability -System architecture (middle ware & database system)	Interview and COTS demonstrations	Nil	Nil
2008	MiHOS	COTS software	Defining acceptable value for each technical goal	-Define evaluation criteria -Searching -Screening -Evaluation -Analyzing	-Functional requirements *Non-functional requirements are not addressed*	Prototyping; product demonstration; and product documents examination	Relying on WSM technique *(doesn't addressing COTS mismatches)*	MiHOS Prototype Tool
2011	UnHOS	COTS software	Nil	-Define evaluation criteria -Search/screening -Evaluation	-Functional requirements *Non-functional requirements are not addressed*	Using prototyping; product demonstration; and documents review	Using AHP and Bayesian Belief Network (BBN) techniques *(don't addressing COTS mismatches)*	UnHOS tool supports

Based on the comparison in Table 2.5, there is a great variety of COTS selection methods being proposed over the last decades. Nevertheless, there are several issues remain inadequately addressed. One of these issues is the lack of providing the accurate decision making technique that handling of the mismatches between user's requirements and the COTS features. Besides saving the system integration time and cost, the mismatch identification is crucial in supporting accurate decision of selecting the appropriate COTS software. For those methods, such as PORE and MiHOS, that have attempted to handle the mismatch issues have a lack to provide an adequate mechanism or a well-defined process to handle them. The PORE does not provide how to deal with unresolved mismatches and the influence of the remaining mismatches on the COTS selection and after selection, while MiHOS method does not consider the cost and time of the mismatches' resolution actions to measure the mismatch level which may give inaccurate decision.

In addition, most of the methods inadequately addressed the non-functional requirements in providing required evaluation criteria for evaluating the COTS software. Some of these methods, such as IusWare and DesCOTS system, only rely on the general software quality models such as the ISO 9126 model to provide quality criteria instead of other important criteria such as vendor and user organization. There are methods that only provide the high-level evaluation criteria without presenting their attributes and metrics such as CRE, STACE, and PECA. According to Scriven (1991) and Zarour (2009), determining the data collection techniques and the evaluation standard (yardstick) that represented by the ideal and lowest values for each criterion are the basic elements in any evaluation process.

Most of the methods (such as OTSO, CISD, CEP, CSSP, and UnHOS) also do not define these elements. Moreover, by not providing any software tool increases the burden on the evaluators, and the complexity of the evaluation and selection process makes the real implementation more difficult.

As mentioned earlier, majority of these methods are less accepted in the industry as practical methods for evaluating and selecting the COTS software. Specific individuals or organization (Mohamed et al., 2008; Ruhe, 2003; Wanyama & Far, 2005; Wanyama & Far, 2008) only applied the few others, such as the CSSP. Thus, a new framework is needed to overcome these limitations and consider the strengths of the current methods, and to bridge the gap between theoretical methods and practice (Sen & Baracli, 2006; Javed et al., 2012; Land & Blankers, 2007).

The following section presents an overview of the main related studies in the literature in order to identify their contributions and limitations in the COTS software evaluation and selection field.

2.3.4 Related Studies

The COTS software evaluation and selection is an extremely important topic in the field of software developments. The popularity of such topic has influenced many researchers to propose related studies. This section presents an overview of most of the related studies by describing it in three main parts: the existing methods for handling COTS software mismatches, the evaluation criteria, and previous empirical studies.

2.3.4.1 Existing Methods for Handling COTS Software Mismatches

Successful COTS software selection process depends on selecting the appropriate COTS software which in turn depends on the accuracy of the decision making. Being able to handle the COTS features and user's requirements mismatches supports the making of effective COTS software selection decision. There have been several attempts in delivering solutions for handling the COTS mismatches such as RCPEP, CARE, PECA, and Win Win spiral. Those methods supported the COTS mismatches identification without providing a mechanism of handling these mismatches (the solutions and constraints) and synthesizing the final decision. Some of the popular approaches are discussed below.

1. **Goal-Based Approach to Guide the Matching Process Using Quality Models**

This approach, proposed by Alves et al. (2005), aims to support the matching process and control the conflicts in CBD. This approach applied the goal-oriented requirements engineering strategy and quality models in order to support the matching between user requirements and COTS capabilities. Some of the matching patterns defined in this approach help decision maker in classifying the matching level between user requirements and COTS capabilities in order to facilitate the following processes: fulfilled, differ, and fail matching.

The concepts from the utility theory are used in this approach to make the decision by measuring the degree of matching between the COTS alternatives and various goals. For instance, using the satisfaction function for operational goals (Sat_{gi}:M

→[0,1] , where M is a set of values may be taken under goal (g)). Each of the matching patterns describes the different degrees of matching between the goals and COTS capabilities. After the matching values and patterns are identified, exploratory scenarios represented by using semi-structured textual form to deal with the mismatches and propose resolution alternatives are suggested.

Among the drawbacks of this approach include; i) focuses on the quality criteria and ignores other kinds of non-functional requirements such as architectural criteria; ii) does not consider the mismatches resolutions constraints (e.g. time and cost) when computing the matching level; and iii) fails to assign weights of goals and perform the final selection according to the identified mismatches.

2. COTS-Aware Requirements and Software Architecture (CAR/SA)

The CAR/SA approach was proposed to support the iterative matching, ranking, and selection of COTS components based on under development system requirements (functional and non-functional) and its architecture (Chung & Cooper, 2004). In this approach, selecting the COTS software goes through a set of steps, as shown in Figure 2.12. Defining the evaluation goals is the first step in this approach; followed by searching the matching of COTS components against the evaluation goals. After that, the COTS alternatives are ranked according to the matching of the evaluation goals.

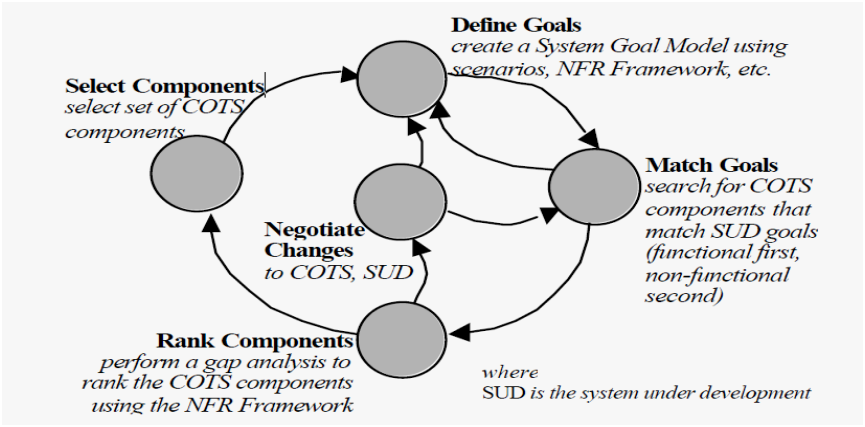


Figure 2.12. The CAR/SA Process

(Chung & Cooper, 2004)

In this context, this approach proposed two groups of matching: i) MAKE that represents the exact matching, one with minor or insignificant matching; ii) HELP that represents close matching with tolerable mismatch. The GA technique is used at the ranking step to identify the matching level of each COTS software component against evaluation goals. The negotiated change step is carried out when there are mismatches between the COTS capabilities and evaluation goals. The result of this process would be the decision to include either the negotiated unimportant goals or modified COTS software.

Unfortunately, this approach suggested using the GA technique to address the COTS software mismatches without explaining how it is applied. This approach also does not suggest any effective solution to handle the mismatches between the evaluation goals and COTS capabilities (Alves et al., 2005). Most importantly, the generation of the mismatches solutions and the identification of the resources constraints are not clearly described.

3. Optimized Mismatch Resolution for COTS Selection called MiHOS

Mohamed et al. (2008) proposed this approach to address the COTS software mismatches by identifying and analyzing a set of mismatches and their resolution actions. This method starts by defining the evaluation criteria as a set of strategic goals (high level) and technical goals (low level) in a hierarchy structure in order to compare them with the COTS features and determine the score that represents the matching level.

The relative weights for the technical goals and the final score of the COTS software are determined and calculated using the WSM technique. This approach is realized through three phases as portrayed in Figure 2.13. The first phase is the problem setting modeling that aims to identify the mismatches, resolution actions, goals weights, and constraints. The second phase is the exploration that aims to explore the solution space by generating set of alternative qualified mismatch solutions (plans). The third is the consolidation phase, which refers to the review of the exploration phase outputs done by decision makers in refining necessary problem.

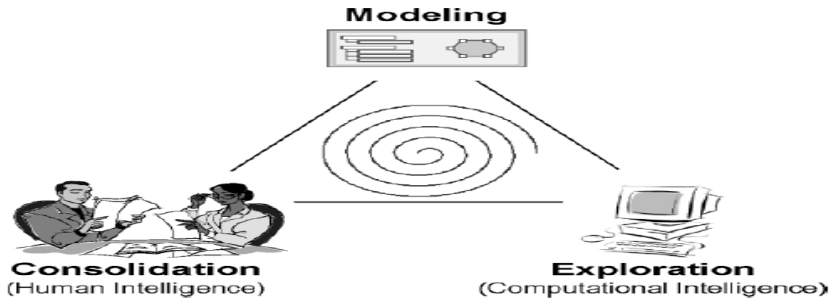


Figure 2.13. The MiHOS Phases

(Mohamed et al., 2008)

Since the MiHOS approach is proposed to address the mismatches during and after the COTS selection, many resolution actions are generated for each mismatch, which make it requires more time, efforts, and cost. This condition is not suitable to be applied within a project that has limited time and budget. The weakness of this approach increases as it relies on the WSM technique, which have many limitations (refer to Table 2.2). Moreover, by not addressing the non-functional requirements and considering the resolution constraints make the technique even worst. This is unacceptable because it may produce inaccurate and unreliable decision.

4. A Process for COTS-Selection and Mismatches Handling

Kiv et al. (2010) proposed a goal-driven agent-oriented approach to facilitate decision-making and the mismatches analysis during and after the COTS selection. This approach aims to take the benefits of goal driven requirements engineering and agent oriented for modeling complex system to improve CBD.

In this approach, the requirements engineering are proceeded using i* (i-star) (Yu, 1997) model with i* goals are refined in a set of agent capabilities. On the other hand, the COTS software is analyzed together with a set of goals, various sub-goals and features. Then, the i* goals can be directly compared against the COTS goals and sub-goals. Then the agent capabilities of user's requirements can be compared to the agent capabilities of the COTS software to address the mismatches on a two level basis.

This following describe the approach as applied in two levels:

i. Macro-level (COTS selection) refers to the level where the organizational environment is analyzed, the potential COTS components are identified, goals are formed, filtering results are searched and evaluated (by addressing the mismatches and the potential resolution actions), and selection decision are made.

ii. Micro-level (mismatches handling) refers to the level where further analysis of the identified mismatches is carried out by defining the realization path (a set of functions the agent should answer to) of unsatisfied goal, and defining the causes of these mismatches and the appropriate resolution action.

The downside of this approach is that it only focuses on the functional requirements. It does not provide the mismatching level measurement and the total fitness score computation of each of the COTS software against the defined goals. Moreover, this approach does not explicitly use any synthesis technique such as the AHP or WSM technique to assign the weights for the goals. It only uses the simple scale (very high, high,…, very low) for prioritizing the goals which is not effective when there are a large number of goals.

In summary, in attempting to handle the COTS mismatches, all of the previous approaches failed to provide effective and complete approach. Table 2.6 provides a comparison between these approaches based on the following criteria:

i. NFR: addresses the non-functional requirements.

ii. IRC: identifies the mismatches resolutions constraints (IRC) (cost, time, effort, risk).

iii. CRC: considers the resolution constraints when computing mismatching level.

iv. SDMP: provides the systematic decision-making process that includes well-defined processes, techniques, and tool support.

Table 2.6

The Comparison of Existing COTS Mismatches Approaches

#	Mismatches Handling Approach	Non-Functional Requirements	IRC				CRC	SDMP
			Cost	Time	Effort	Risk		
1	Goal-based approach to guide the matching process using quality models	Focused on the functional requirements and ignore the non-functional requirements	√	x	x	√	Nil	Nil
2	CAR/SA	Focused on the quality aspects and ignore other aspects	x	x	x	x	Nil	Nil
3	MiHOS	Addressed the mismatches with the functional requirements and ignore the non-functional requirements.	√	x	√	√	Nil	Nil
4	A process for cost-selection and mismatches handling	Focused on the functionality requirements and ignore the non-functional requirements	x	x	x	x	Nil	Nil
√: addressed							X: not addressed	

As a depicted in Table 2.6, most of the previous COTS mismatches approaches addressed the mismatches at the functional requirements and ignored the vital role of the non-functional requirements. Furthermore, the identification of the mismatches between the COTS software and user's requirements is not enough to provide

accurate decision. Higher accuracy can be achieved if the mismatches resolution actions and their constraints are taken into consideration. For example, the low cost mismatch resolution with high risk will negatively impact the final system.

In this context, only two approaches partially considered the mismatches resolutions constraints, the MiHOS and goal-based approach. Both approaches only look into the costs and risks factors without including the required time and effort for each mismatch resolution. They also neglected the importance role of these constraints in selecting the fitness COTS software especially when calculating the mismatch level between the COTS software and user's requirements. Furthermore, all of these approaches do not provide a well-defined and systematic decision making process in terms of defining the criteria, searching and filtering the COTS alternatives, and collecting the data. Hence, a well-defined and efficient process for handling COTS software mismatches and making accurate decision is still missing in the COTS software evaluation and selection process.

2.3.4.2 Evaluation Criteria

The evaluation criteria refer to the facts or standards by which the fitness of the COTS software is assessed. Defining the evaluation criteria is performed through a straightforward decomposition of the non-functional requirements starting from high level requirements until producing pieces of well-defined measurement information (Comella-Dorda et al., 2004). According to Sommerville (2011), the non-functional requirements have a critical role in software evaluation than functional requirements. Since the evaluation criteria have the essential and decisive role to measure the

software quality in software industry (Cai et al., 2011, Lloyd, 2005; Chung & do Prado Leite, 2009), the establishment of the evaluation criteria is a very important task for understanding, evaluating, and selecting the best COTS software (Lewis & Morris, 2004).

Many evaluation criteria have been used to develop several models for measuring, discussing, and rating software quality (Cai et al., 2011; Rawashdeh & Matalkah, 2006). Most of the existing models are proposed to evaluate the general software quality such as the McCall's, Boehm's, and ISO 9126 models. None of them are dedicated to handle specific characteristics of the COTS software (Alvaro et al., 2010; Bertoa & Valecillo, 2002; Kalaimagal & Srinivasan, 2010/b; Rawashdeh & Matalkah, 2006). Even though there are models that have been developed exclusively for evaluating the COTS software, the appropriate supporting evaluation criteria are still inadequate. For example, most of the models have never included the vendor supportability and reputation (Cai et al., 2011; Kalaimagal & Srinivasan, 2008; Lloyd, 2005). Thus, it can be concluded that in reality, the COTS software evaluation is still in its immature stage.

The similarities of the related models are described next. In case the models have not been given name by their authors, the name of the first author or some time the title of the publication were used to name these models.

Kunda and Brooks have presented the work on the social-technical criteria in 1999 to support the evaluation of the COTS software. The proposed criteria have been

determined through conducting a set of interviews with developers in seven companies. These criteria consist of technology factors, functional characteristics, product quality characteristics, and social-economic factors. Although this work proposed a diverse set of evaluation criteria (technical and non-technical), the researchers does not present the measurement details of the studied organizations and their potential stakeholders. At the same time, the proposed criteria do not compromise with other important criteria for the COTS software evaluation such as portability and popularity. By just relying on a small number of companies does not often provide reliable results.

Bertoa's Quality Model was the earliest quality model developed for the COTS software evaluation by Bertoa and Valecillo (2002). This model attempted to identify a set of attributes that can be estimated based on the information from the COTS vendors to evaluate and select COTS software. The quality characteristics and sub-characteristics are similar to that of the ISO 9126 model with a simple change of adding the compatibility sub-characteristics under the functionality component. This model also proposed a variety of attributes associated with the sub-characteristics, presented with various kinds of measurements such as percent, integer, ration, and time. Although presenting a good description of the quality characteristics, sub-characteristics, and metrics, this model still has many weakness points. By just making a minor modification (removing six sub-characteristics) to the characteristics of the ISO 9126 model, the Bertoa's Quality Model is still viewed as not suitable to be applied in the COTS domain because of the general evaluation implementation. Furthermore, this model does not address other important characteristics such as

popularity and reusability. Finally, this model is regarded as incomplete due to the failure of carrying out an experiential assessment (Sharma et al., 2008).

The COTS Quality Model (C-QM), proposed by Kim and Park (2003), is defined based on a set of COTS software characteristics such as the common functionalities, required interface for customization, large granularity reusability through component interface, and component reference model dependency. These characteristics are associated with four quality factors at the top level, and twelve criteria at the low-level. However, this model only presented a few factors to evaluate the COTS software and neglected other quality factors that have important role such as portability, reliability, and interoperability (Cai et al., 2011; Kalaimagal & Srinivasan, 2010/a). It also ignored other sources of factors that are important to distinguish between the COTS alternatives and help select the fitness COTS software such as vendor and operational environment attributes which are stressed by many studies conducted by (Beus-Dukic, 2000; Carvall & Franch, 2006; Elanchezhian et al., 2010; Kunda & Brooks, 1999).

The non-technical criteria Quality Model, proposed by Carvall and Franch (2006), were an extension of the ISO/IEC 9126 model. The model highlighted the importance of non-technical criteria (i.e. supplier's characteristics) beside the technical criteria in minimizing the hindrance of the COTS selection processes. The ISO/IEC 9126-1 quality standard was used to evaluate the technical criteria as it represents the most widely standard applied by the software engineering community. The proposed non-technical criteria were sorted in ISO/IEC 9126 tree-like structure

where the high level of the hierarchy consists of three characteristics: supplier, cost, and product. These characteristics were decomposed into fifteen sub-characteristics. Those sub-characteristics were further decomposed into over two hundred attributes.

The reliance on the ISO/IEC 9126 model hinders the effectiveness of the non-technical criteria Quality Model because the initial model is only suitable for evaluating software with general functionalities. This downside is similar to the other general software quality models that ignore the special characteristics of the COTS software (Alvaro et al., 2010; Kalaimagal & Srinivasan, 2010/a). Moreover, the proposed non-technical criteria were decomposed into a huge number of attributes (200 attributes) that increases the time and efforts which are limited in the evaluation process. In addition, the technical criteria (ISO/IEC 9126) and non-technical criteria have not covered other important characteristics for evaluating the COTS software such as popularity, organization characteristics, and vendor stability characteristic.

Rwashdeh and Matalkeh (2006) developed the Quality Model for COTS Components by providing a set of quality characteristics to evaluate the COTS software. Using this mode, they attempted to determine the type of stakeholders for each high level of quality characteristics. This model adapted the ISO 9126 quality model to identify the quality characteristics and sub-characteristics. They have added a new high-level characteristic called manageability that was broken down into quality management sub-characteristic. In addition, the compatibility and complexity had also been added as sub-characteristics. This indicates that there is no significant improvement compared to the previous models. This model had removed the

portability characteristics and its sub-characteristics that had since been used and considered by these researchers (Alvaro et al., 2010; Bohem & Abts, 1999; Dean & Gravel, 2002; Gill, 2006; Kunda, 2003). They believed that portability and its sub-characteristics are important to identify the interaction of the COTS software within different environments. In the same way, the fault tolerance and stability sub-characteristics had also been eliminated despite the importance of them in COTS evaluation. Finally, the quality attributes and measurements covered in this model remained ambiguous, especially, with the introduction of the new high-level manageability characteristic and its quality management sub characteristic (Kalaimagal & Srinivasan, 2008).

The quality model for Component-based systems was proposed by Sharma et al. (2008) for CBS evaluation. It aims to assess the quality of any software component before the final system integration. It also can be used for estimating the efforts needed for achieving the desired value for any quality characteristic. The basis of this model is the ISO 9126 model. The only difference is the additional and removal of some of the characteristics and sub-characteristics such as complexity, traceability, and reusability. This model does not use all of its quality characteristics when evaluating software. Only those required and important characteristics and sub-characteristics are used to estimate the software component. The AHP technique is also used for assigning weights in order to identify the important characteristics and sub-characteristics that are related to software.

The fact that this model relies on the ISO 9126 model by presenting similar quality characteristics and sub characteristics with simple modification (added five new sub-characteristics), makes the model still general to be applied in the COTS domain. Moreover, this model also suffers from the bias of selecting criteria that are importance from other model to evaluate particular application.

The Q'Facto 12-Quality Model for COTS Components was proposed by Kalaimagal and Srinivasan (2010) as the recent quality model to evaluate the COTS software (Kalaimagal & Srinivasan, 2010/b). The model consists of three levels, with the first contains twelve quality factors, twenty eight quality criteria in the second, and forty eight quality measures in the third level.

Evidently, this model has more advantages compared to the previous quality models since it was developed based on the ISO 25000 quality standard, which is a higher version of the ISO 9126 quality model. Furthermore, this model highlights the importance of interoperability and security quality characteristics specifically in the COTS domain by putting them as quality factors. This differs from that of the previous quality models, which considered them as sub-characteristics. Finally, this model provides a set of quality measures for the quality criteria that are missing in most of the previous quality models.

However, it has also been observed that the Q'Facto12 quality model suffers from several weaknesses. Despite providing additional quality factors and criteria, many other factors (i.e. the maturity of software and the popularity in specific domain)

should be considered when estimating the COTS software. This is what lacking in this model and other previous models. Similarly, this model takes a lot of time and effort in computing the quality factor measurements. Furthermore, this model was developed only to be used by end users, while the quality measures that can be performed by the end user are limited. In addition, this model was just recently developed and its validity has yet to be approved by the industry.

Discussion and Comparison

The appropriate evaluation criteria model for evaluating the COTS software must consider and fulfill all the stakeholders' requirements. Therefore, based on the previous survey and detailed analysis from the literature regarding the COTS evaluation criteria and models, the described models are still not comprehensive and complete. There are many related issues raised and still inadequately addressed to provide the most appropriate model for evaluating the COTS software. These are discussed as follows:

The models discussed above are good in handling the quality criteria for the COTS software evaluation. Nevertheless, these models are still lacking in providing adequate quality criteria. A good COTS software evaluation model should consider all of the required evaluation criteria that help to distinguish between the COTS software products and to trace their reliability. Most of the existing models were derived from the general software quality models (e.g. ISO 9126) that provide general quality criteria that are not appropriate for the COTS software evaluation (Alvaro et al., 2010; Kalaimagal & Srinivasan, 2010/a).

The nature of the COTS software make it necessary to estimate additional important criteria related to the vendor that developed, upgrade, and support it, as well as the criteria related to the operational environment that used this software. These kinds of criteria are always discarded by most of the previous models (Alvaro et al., 2010; Carvall & Franch, 2006). In addition, Thapar et al. (2012) suggested that a good evaluation results could only be achieved by following a well-defined and reliable evaluation process. Therefore, to become successful, a COTS criteria evaluation model needs to be managed and conducted formally by applying the right efforts, planning, technique, and guideline.

It is a known fact that quality is best perceived by user's satisfaction. Therefore, each group of criteria evaluation model should be classified for different kinds of stakeholders that are responsible of providing the required information criteria. Most of the previous models do not include the involvement of stakeholders and determine the source of data collection. Regarding the fairness quality validation, most of these models were just theoretically proposed and not evaluated through industrial scenario such as the quality model developed by Rwashdeh and Matalkeh (2006) and Q'Facto12 by Kalaimagal and Srinivasan (2010/b). Thus, the efficiency of evaluating the COTS software using these models remains unknown (Alvero at el., 2005; Carvall & Franch, 2006; Kalaimagal & Srinivasan, 2008). Furthermore, some of these models have subjectivity in the evaluation due to the lack in defining the metrics for measuring the provided evaluation criteria. It is essential to provide the metrics in the model and their usage should also be described in sufficient details (Thapar et al., 2012).

All of the previous issues pose the challenges in providing the appropriate model for evaluating the COTS software. In order to estimate the previous models against these issues, Table 2.7 is established to show the purpose of these models against the set of factors that are summarized as the main issues in the previous discussion.

Table 2.7

Evaluating the Existing Models of the COTS Software Evaluation

Models / Factors	**Social-Technical Criteria** *(Kunda & Brooks, 1999)*	**Bertoa's Quality Model** *(Bertoa & Valecillo, 2002)*	**C-QM** *(Kim & Park, 2003)*	**Non-Technical Criteria Quality Model** *(Carvall & Franch, 2006)*	**Quality Model for COTS** *(Rawashdeh & Matalkeh, 2006)*	**Sharma's Quality Model** *(Sharma, 2008)*	**Q'Facto 12-Quality Model** (*Kalaimagal & Srinivasan, 2010/b)*
Evaluation criteria focused	Quality characteristics and some of non-technical characteristics	Quality characteristics	Quality characteristics	Quality characteristics and some of non-technical characteristics	Quality characteristics	Quality characteristics	Quality characteristics
Data gathering technique	Brainstorm and documents review techniques	Not included	Not included	Not included	Not included	Not included	Not included
Data synthesis technique	AHP technique	Not included	Not included	Not included	Not included	AHP technique	Not included
Evaluation process	Provide set of process such as requirements definition, and identifying COTS software	Not included	Not included	Not included	Not included	Not clear	Not included
Determining the target stakeholders	Stakeholders (without specification)	Software architects, designers	Undetermined	Undetermined	End user, analyst, quality assurance, business owner, and project manager	End user	End user
Practically validation	Through case study	Not included	Not included	Not included	Not included	Through Case study	Not included
Subjectivity in the evaluation	Does not provide the attributes or metrics	Provided set of attributes and metrics	Provided set of attributes and metrics	Provided set of attributes and metrics	Does not provide the attributes or metrics	Provided set of attributes and metrics	Provided set of attributes and metrics
Levels & number of criteria	7 main factors 29 sub-factors	6 characteristics 16 sub-char. 44 attributes	4 characteristics 12 sub-char. 12 metrics	3 characteristics 15 sub-char. Over 200 attributes	6 characteristics 17 sub- char.	6 characteristics 26 sub-char.	12 quality factors 28 quality criteria 40 quality measures
Based model	ISO 9126:1991	ISO 9126:2001	CORBA Component	ISO/IEC 9126-1	ISO 9126:1991	ISO 9126:2001	ISO 25000

		(for technical factors)		Model (CCM) & ISO 9126				
Evaluation criteria modifications	Main-char	Technology factors (+) Business issues (+) Customer capability (+) Marketplace variables (+) Vendor capability variable (+)	-Usability (c)	Performance (+) --- Portability (-) Efficiency (-) Usability (-) Reliability (-)	Supplier (+) Cost (+) Product (+)	Manageability (+) --- Usability (c)	No modification	Context Coverage in Use(-) Efficiency in Use(-) --- Freedom from Risk(c)
	Sub-char	Performance, Framework and architecture style, Interface standard, Concepts of evaluation and versioning, Development environment (+) --- Integration, upgrade, licensing, product, support, technology, and training costs (+) --- Customer expectations, experience, and organizational policies and politics (+) --- Market trends, product reputation (+) --- Vendor stability, reputation, certification, and availability of training and support (+)	Compatibility (+)	Commonality (+)	Organizational structure (+), positioning and strength (+), reputation (+), services offered (+), support (+)	Compatibility (+) Complexity (+) Quality-management(+)	Reusability(+) Complexity (+) Scalability (+) Trackability (+) Flexibility (+)	Self-contained(+) Generality(+) H/S Independence(+) Locatability(+)
			Fault tolerance (-) Stability (-) Analyzability (-) Installability (-) Conformance (-) Adaptability (-)	Modularity (+) Customizability (+) Comprehensiveness (+)	Licensing schema (+) Licensing costs (+) Platform cost (+) Implementation cost (+) Network cost (+)	Fault tolerance (-) Stability (-) Analyzability (-)		Capacity(-) User Error-Protection(-) User interface-aesthetics(-) Accessibility(-) Maturity(-) Availability(-) Non-repudiation(-) Accountability(-) Authenticity(-) Analyzability(-) Modularity(-)
			Learnability (c) Understandability (c) Operability (c)	Standard conformance (+) Reference model conformance (+)	Stability (+) Ownership (+) Deliverable (+) Parameterization (+) Guarantee (+)			

+ : this model has been added this characteristic or sub-characteristic
- : this model has been removed this characteristic or sub-characteristic
C: this model has been changing the meaning of this characteristic or sub-characteristic to specific domain.

Table 2.7 denotes the comparisons between all of the previous models, which have not succeeded in providing appropriate model for the COTS software evaluation. An appropriate evaluation model should offer the required criteria that meet all the functional and non-functional requirements. In addition, the model would be better off if supported by suitable evaluation process, technique, guidance, and documentation to facilitate the final decision-making. Related information of the participated stakeholders will simplify the data collection and the COTS software estimation activities. The evaluation criteria measurements should be explicitly defined by providing their relevant metrics and at the end, the model should be fairly validated by independent party or expert academicians (Alvero at el., 2005; Alvaro et al., 2010; Kalaimagal & Srinivasan, 2010/a; Thapar et al., 2012).

In this perspective, several pieces of work provide for ways to distinguish among different kinds of non-functional requirements such as work proposed by Sommerville (2011), Mellor (1992), Carvall and Franch (2006), and ISO/IEC 9126. Table 2.8 shows these works and their proposed classifications.

Table 2.8
Non-Functional Requirements Classifications

The Study	The Proposed NFRs Classification	Resources
ISO/IEC 9126 classification	1) Quality in use *(e.g. effectiveness, productivity, and satisfaction)* 2) External quality *(e.g. functionality, reliability, and usability)* 3) Internal quality *(e.g. maintainability and portability)* 4) Process quality *(e.g. sustainability and performance)*	ISO/IEC (2001)
McCall's classification	1) Product revision *(e.g. testability and flexibility)* 2) Product transition *(e.g. portability and reusability)* 3) Product operations *(e.g. reliability and usability)*	Mellor (1992)
Sommerville classification	1) Product requirements *(e.g. usability, reliability, and portability)* *2)* Organizational requirements *(e.g. delivery and implementation)* 3) External requirements *(e.g. interoperability and ethical requirements)*	Sommerville (2011)
Carvall's classification	*1)* Supplier *(e.g. support, service offered, and reputation)* *2)* Cost *(e.g. licensing schema, and licensing cost)* 3) Product *(e.g. stability, ownership deliverable, and customization)*	Carvall and Franch (2006)

Regarding the evaluation of the COTS software, Beus-Dukic (2000), and Kunda (2002) stressed that the specialist of COTS software would bring in additional important non-functional requirements which include the architectural, domain, user organization (operational environment), and vendor organization requirements (Figure 2.14).

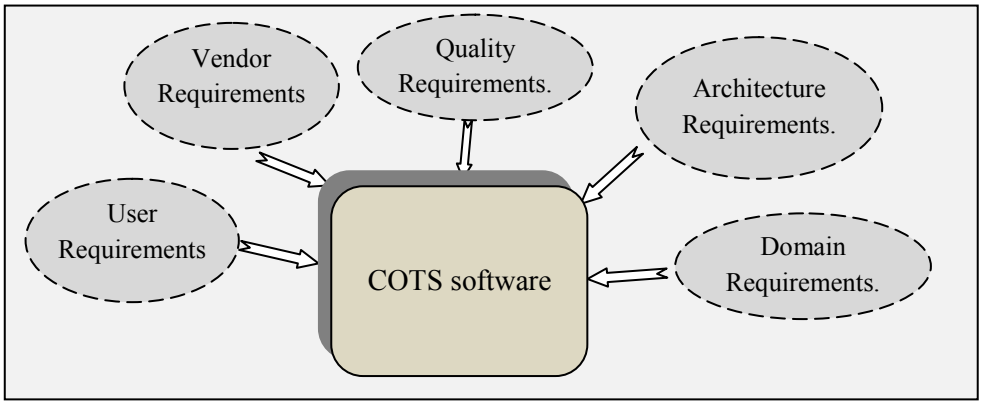


Figure 2.14. The Non-Functional Requirements for COTS Software (Adapted from Beus-Dukic, 2000)

However, the previous NFRs classifications (Table 2.8) do not consider the nature of COTS software evaluation. Precisely, they just focused on the product quality as the classifications proposed by McCall and ISO/IEC 9126 quality models, while the Carvall and Franch and Sommerville classifications mentioned about the organizational, cost and supplier requirements in implicitly way. In fact, all of them ignored the importance of the domain and system architectural requirements. Consequently, the previous classifications have insufficiently addressed all the NFRs that are required for COTS software evaluation.

Therefore, the proposed NFRs classification in Figure 2.14 is the most suitable and comprehensive comparing with other classifications to address the nature of the COTS software by providing the required evaluation criteria for evaluating the COTS software (Crnkovic et al., 2005; Kaur & Mann, 2010).

Consequently, the non-functional requirements classification (Figure 2.14) was used in this research to identify and classify the COTS evaluation criteria. As well as the important evaluation criteria for the COTS software were proposed in this research, it is concentrating on the identified issues that have not been addressed previously to develop new model for COTS evaluation criteria. These criteria were identified and decomposed based on several previous works such as the work by Bertoa and Valecillo (2002) and Kalaimagal and Srinivasan (2010/a). More details about these evaluation criteria are covered in chapter five.

2.3.4.3 Related Empirical Studies

Many studies have proposed ways of handling the evaluation and selection of the COTS software. However, most of these studies are only concerned with the researchers' view and neglect the developers or practitioners' view. Most of these studies are still in the theoretical part and have not been accepted in practice. For this reason, the empirical study is required to investigate the state-of-practice in order to provide methods, theories, and techniques that more are appropriate in the practice.

In this context, a few researchers have embarked on the empirical studies related to the COTS software evaluation and selections. Among the studies is the one

conducted by Kunda (2002) as an attempt to contribute towards the improvement of the CBS by reducing its risk and cost. The study that emphasized on enhancing the COTS software evaluation and selection process involved SMEs and software houses in Zambia and United Kingdom. Another study was the one carried out by Dagdeviren et al. (2005) to address the problem of identification, evaluation, and selection of the COTS and Open Source System (OSS) components by looking at how the market of COTS/OSS components influences in the evaluation and selection. Li et al. (2005) conducted a survey approach by applying semi-structured interview involving 13 Norwegian companies that had CBS projects in order to investigate the state-of-practice of the COTS software selection processes. Another empirical study conducted by Land et al. (2009) using web-based questionnaire was to investigate how CBS software engineers support the reuse components in practice.

All of the previous studies' objectives, used approaches, main findings, and limitations are shown in Table 2.9.

Table 2.9

The Related Previous Empirical Studies

The Study	Main Objectives	Approach	The Main Findings	Limitations
Kunda (2002)	➢ To find out the current practices (like procedures, methods, and approaches for building CBS), risks, and the benefits of CBS form UK and Zambia	survey (self-administrative questionnaire)	• The most important benefit of CBS in UK and Zambia is reducing the software development cost. • The main obstacle is the lack of appropriate trained human resource. • There was problem in the process of COTS software selection in UK and Zambia. • They focus on the quality evaluation criteria and ignore other.	▪ It focused on the developing countries, Zambia. ▪ It focused on understanding of CBS like benefits, risks, and approaches.
Dagdeviren et al. (2005)	➢ To recognize any decisive factors in order to support the effectiveness of task identification and selection.	survey approach(questionnaire)	• The majority of providers' websites did not select the reflective names for their products. • They also did not provide the explicit functionality. • The providers did not determine the hardware and software that required to success integration of their COTS/OSS products.	▪ It only focused on the part of software products marketing. ▪ Estimating COTS software is different than estimating OSS software.
Li et al. (2005)	➢ To investigate the state-of-practice for the processes of selection COTS software in Norwegian companies. ➢ To explore the reasons behind using them and their challenges, or the reasons for not using these processes.	Survey approach (semi-structure interview)	• The COTS software selection processes are not limited to the formal processes.	▪ It is based on few and non representative of the sample in Norway. ▪ Most of COTS software components in this survey support the minor part of the functionality of the whole system, and thus works as a spurt part.
Land et al. (2009)	➢ To know how and to what extent components are verified in isolation, and how the software components are evaluated and selected.	the web-based questionnaire	• The software components-reuse make the design decisions more easy. • The requirements changes in practices still not inconclusive, while the verification is not done in adequate way either reuse the software components or not. • The filtering is necessary to component candidates in high-level evaluation phase using the information and documentation of component. • Many organizations not yet applied the practice to iterative the components selection with requirements elicitation as recommended by the method PORE and CRE. • Many organizations and project selected software components without proper evaluation.	▪ It is the very low of response rate, which decrease the reliability of the findings. ▪ Since the respondents of this survey are anonymous, these responses do not indicate how many organizations were represented. ▪ The invitation was sent to specific list of email and encouraged the recipient to further spread the invitation, which make neither identify the response frequency, nor which exactly the organizations were represented.

In conclusion, all of these studies provide a good information related to the reuse software components in general and the COTS software selection in practice. Nevertheless, some of these studies relied on the small sample size such as the study conducted by Li et al. (2005), which involved 13 organizations only. This reduces the reliability of the findings. Moreover, these studies have not addressed various issues that represent significant challenge for the practitioners such as the COTS mismatches, decision-making process, and evaluation criteria. Most of the studies only addressed the general issues related to the components-based systems development. Therefore, it is important to conduct a new survey that includes the required practitioner information. This information will relate to the challenges associated with the COTS evaluation and selection. This is important to support the applicability and acceptability of a new framework development for the COTS software evaluation and selection in industry.

2.3.5 Issues and Challenges in COTS Software Evaluation and Selection

Based on the previous discussion, successful CBD depends on the successful of the COTS software evaluation and selection. This means that the COTS software evaluation and selection plays a vital role in CBD. However, the implementation of this process is still facing many challenges and risks. These include the following:

Lack of well-defined, systematic, and repeatable COTS software evaluation and selection process: this means that the process of COTS software evaluation and selection should include and define all the required processes, activities, tasks, techniques, and software tools for evaluating and selecting COTS software which make it easy to perform many times. Since the COTS software evaluation and

selection process is a non-trivial task most organizations are under pressure because they usually perform the COTS software evaluation and selection in an ad-hoc manner (Javed et al., 2012; Pande et al., 2013; Vijayalakshmi et al., 2008). The evaluators may not have enough time, and experience to plan, and carry out the COTS software evaluation and selection. Therefore, if they choose an inappropriate method, wrong decision may contribute to the project failure. When the COTS software evaluation and selection process is not defined, the particular activities will be reinvented each time and eventually resulted in inconsistent performance. At the end, to learn from previous cases can be difficult (Jingyue et al., 2009; Kunda, 2003; Vega, 2006).

Ineffective evaluation criteria: According to Montecillo (2009), evaluation criteria are described as a challenging task. Alves et al. (2001); Land et al. (2008); and Asghar and Umar (2010) highlight the lack of considering the non-functional requirements that raise the risks of the COTS failure and the cost of the final system development. The non-functional requirements play important roles in addressing the quality issues of the COTS software, which involves usability, stability, portability, security, and others (Alves & Castro, 2001; Ye & Kelly, 2004).

Lack of handling the COTS mismatches: Addressing the mismatches between user's requirements and the COTS features supports the accuracy of the best fitness selection decision (Ibrahim et al., 2011; Kiv et al., 2010; Mohamed et al., 2008). The handling of the COTS mismatches during the COTS evaluation and selection reduces the time, effort, and development, integration and testing costs. However,

the existing methods for COTS selection failed to deal with the COTS mismatches, which means that they provide inaccurate selection decision which negatively impact the final system (Jadhav & Sonar, 2011; Fahmi & Choi, 2009).

According to Gupta et al. (2012); Jadhav and Sonar (2011); Mohamed (2007); and Ruhe (2003), the COTS evaluation and selection process involves a multi-criteria decision-making, in which the COTS mismatches and evaluation criteria play essential role in selecting the appropriate COTS software. Moreover, the success of the COTS software evaluation and selection process depends on two main issues: the evaluation criteria to evaluate the COTS software alternatives and the accuracy of the decision-making process in selecting the fitness of the COTS software alternatives. Therefore, the non-functional requirements problem was included in this research to provide appropriate evaluation criteria for the COTS software alternatives evaluation. The COTS mismatches problem was also addressed in order to provide accurate decision-making process for selecting the fitness of the COTS software by offering a well-defined, systematic and repeatable method for evaluating and selecting the appropriate COTS software.

2.4 Summary

This chapter presents the COTS software evaluation and selection process in detail. It also includes the overview of the COTS software; its advantages and disadvantages; and its importance. The detailed descriptions of the COTS software evaluation and selection process are presented. The survey on the existing COTS software evaluation and selection methods is also discussed. The main results show

that majority of these methods have a lack to address the mismatches between user's requirements and COTS features. The models do not provide comprehensive criteria for evaluating the COTS software, which resulted in an inaccurate final decision the COTS software fitness.

The evaluation theory is also presented and discussed as the main theory that used to construct the framework of the COTS software evaluation and selection. In addition, the common COTS selection processes and activities are also described. Besides addressing and analyzing the MCDM and the mismatches handling methods, other the related studies of the COTS evaluation criteria are discussed. As the empirical study is an essential study to elicit the current practice in this research, this chapter presents the previous most related empirical studies of the COTS software evaluation and selection from the literature. Finally, this chapter presents the main issues and challenges of the COTS software evaluation and selection.

CHAPTER THREE
RESEARCH METHODOLOGY

3.1 Introduction

The primary purpose of this research is to propose a new framework to support and improve the COTS software evaluation and selection process. The evaluation theory is used to identify and integrate the required components of the COTS software evaluation and selection. Other necessary actions are to identify the main processes, activities, techniques, and common evaluation criteria; improve the decision-making process by handling the COTS mismatch issues; and evaluate the feasibility and applicability of the proposed framework using case study approach.

This chapter presents the research methodology to answer the research questions and to achieve the research objectives. A clear definition of the research design starts the discussion of this chapter, followed by the descriptions of the procedures or methods performed in delivering the expected research results.

3.2 Research Design

This research applies a deductive approach (Trochim, 2006), which begins with a general idea (such as theory, principles, and concepts) and moves to a more specific conclusion. This is also referred to as "top-down" approach and suitable for model development especially when theories or concepts are derived from literature and empirical findings. The proposed model will then be applied and evaluated in real environment (Bryman & Bell, 2007). This methodology consists of four phases: 1)

theoretical study, 2) empirical study, 3) framework development, and 4) framework evaluation. See Figure 3.1.

These phases are used to develop new framework for the COTS software evaluation and selection. Each phase consists of goal achievement, set of inputs, activities of goal achievement, and set of deliverables. The following sections explain in detail these phases.

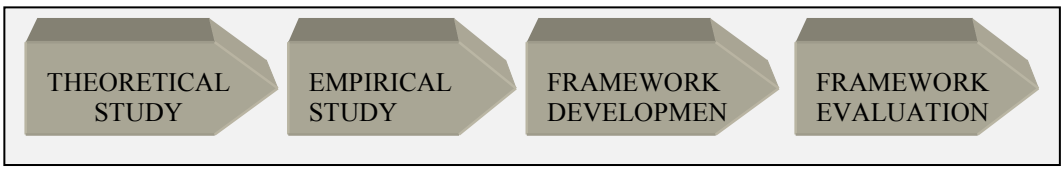


Figure 3.1. Research Methodology

3.3 Phase One: Theoretical Study

In this study, literature in the related research area are reviewed in depth using several resources such as journals, papers, books, documents, proceeding, and other academic research. This study aims to identify the state-of-the-art of the COTS software evaluation and selection. In order to achieve the aim, several activities are conducted such as identifying the main problems, analyzing the existing methods of COTS evaluation and selection, and determining the common processes, activities, and techniques from previous studies. Figure 3.2 illustrates the goal, inputs, activities, and deliverables of the first phase.

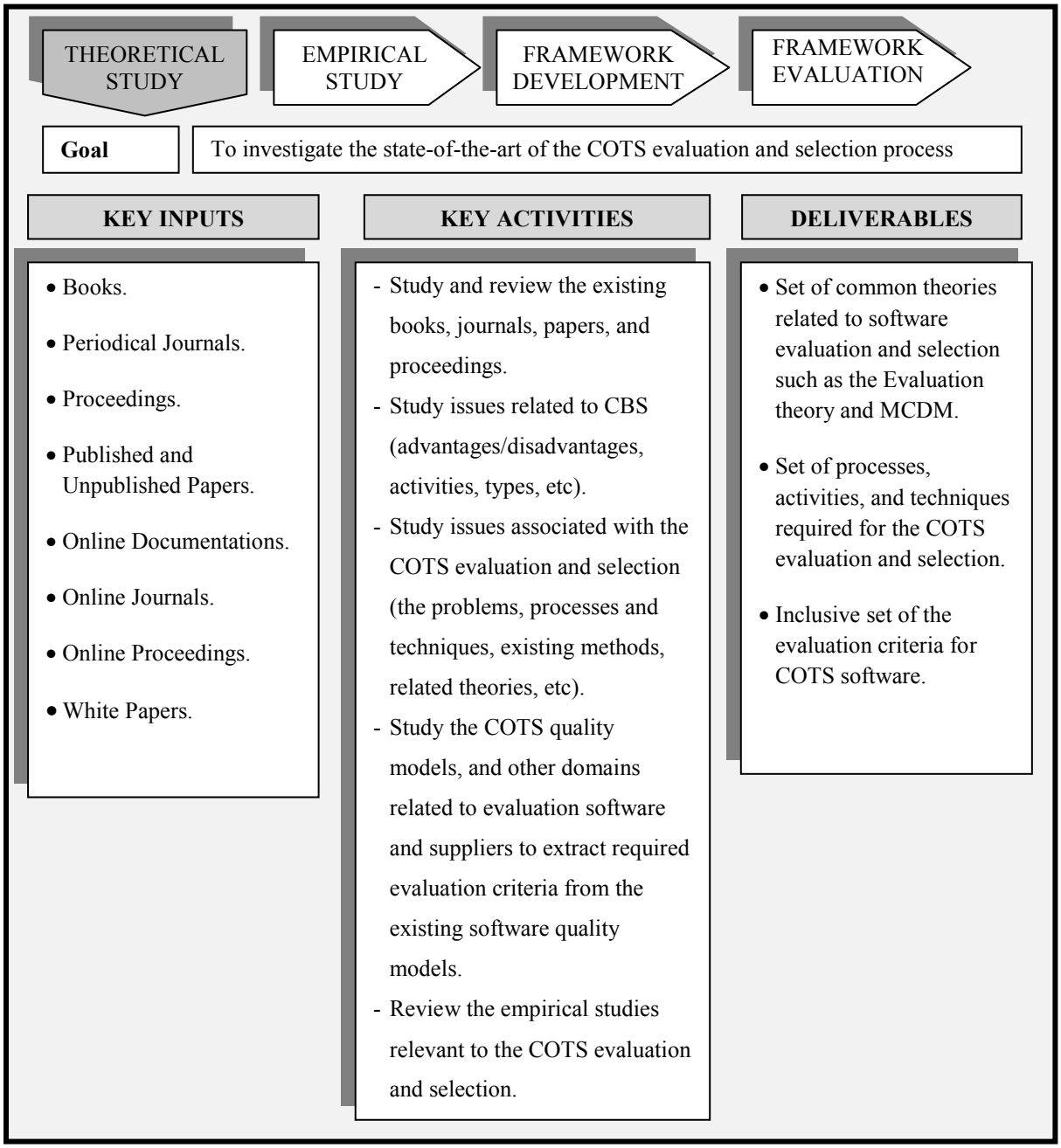


Figure 3.2. Theoretical Study

3.4 Phase Two: Empirical Study

An empirical study is oriented towards discovering, describing, validating, and holistic understanding of processes, activities, and characteristics of current phenomenon in order to extend or develop new theories or methods to improve the

current practice. In this context, the empirical study is performed in this research to investigate the current practices and problems of the COTS software evaluation and selection in Jordanian organizations. To do that, this study aims to: i) verify the current practices of the CBS, ii) determine the benefits and risks associated with building systems from the COTS software, iii) identify the main problems and challenges related to the COTS software evaluation and selection, iv) identify the important processes, activities and techniques for selecting COTS software, v) identify the importance of the NFRs for evaluating and selecting the COTS software, and vi) identify the common evaluation criteria for evaluating the COTS software.

The explanation of the activities carried out to achieve these objectives is laid out in the next sections.

3.4.1 Study Approach and Data Collection Instrument

According to Saunders et al. (2007), selecting the approach depends on the aims and objectives of study. The survey approach was used as data gathering and analysis approach because it useful and powerful in finding answers to research questions through data collection and subsequent analysis and better method to measure awareness and opinions (Sekaran & Bougie, 2010; Yin, 2003). The questionnaires was used as an instrument to collect the data because it has several advantages such as cost effectiveness; ease to analysis the data, coverage a wide area, and also it supports a high degree of secrecy (Kirakowski, 2000; Sekaran & Bougie, 2010).

3.4.2 Sample Procedure

The target population for this study is the organizations that are using the COTS software in Jordan. Unfortunately, the exact number of those organizations is unknown. Even the Ministry of Industry and Trade or other official enterprises in Jordan do not have such information. Due to this limitation and the fact that not all organizations in Jordan are using the COTS software to develop their systems, this study has decided to choose its organization participants through convenience sampling. This is considered as the most appropriate sampling technique because it enables information to be collected from the population members who are conveniently available. Furthermore, the technique is fast, easy, efficient, and inexpensive, which make it suitable for initial or exploratory study (Fox, 2010; Sekaran, 2003). A sample of 200 participated organizations should be convincing enough for this study. This corresponds to Bailey's (2008) recommendation that the sample size of 100 is sufficient and Roscoe's (1975) rule of thumb, sufficient sample size is between 30 and 500. The minimum sample size of 30 is acceptable for statistical analysis (Fisher, 2007; Sekaran, 2003). In this case, the 200 organizations that have been identified and selected randomly based on the 2010 Jordanian Ministry of Industry and Trades report 2010 should be acceptable. More details about the response rate are described in section 4.4.

3.4.3 Instrument Development

The questionnaire was developed based on the guideline proposed by Gay et al. (2006). It is important that the questionnaire is attractive and brief, contains only

items that relate to the study's objectives, collects demographic information as necessary, focuses on items based on single topics or ideas, defines and explains ambiguous terms, words the questions clearly, avoids leading questions, organizes items from general to specific, keeps items and response options together, and finally be pilot tested. Also, careful attention is given to the length of the questionnaire, as well as the length, content, order, and type of individual questions. The questionnaire was developed based on adapting the questionnaires from reviewing the past empirical studies related to COTS evaluation and selection and CBS such as Kunda (2002) and Gerea (2006) (more details in Appendix A). The questionnaire consists of four sections: (i) demographic data; (ii) current practices of CBS; (iii) current practices of COTS evaluation and selection; and (iv) COTS evaluation criteria, see section 4.2.

3.4.4 Pilot Test

The pilot questionnaire was administered at 13 organizations in Jordan. The pilot test was conducted to give respondents the opportunity to comment on the questions asked and explain their responses. The major purposes of the pilot testing was to ensure the validity (measuring the phenomena intended), completeness (include all required items), and readability (avoid misinterpret the questions) of the developed questionnaire. This helped to improve the questionnaire by making further modification, removing irrelevant questions, and determining the timing for answering the questionnaire. More details of this are presented in section 4.3. The final draft of the questionnaire is found in Appendix B.

3.4.5 Survey Execution

The survey execution process starts by preparing the list of participated organizations. This is essential to keep track of these organizations during this activity and to avoid duplication while distributing the questionnaire. The questionnaire was delivered using the mail-back survey because it provides more opportunity to reach broader audience and it is more convenience for the respondents. The questionnaire was also hand-delivered because it presents opportunity for personal interaction with respondents and it receives a greater response rate. Four weeks were given for each organization to answer the questionnaire so that the response rate can be maximized. A reminder was sent to those who failed to complete the questionnaire after the given period.

3.4.6 Data Analysis Procedures

In the data analysis activity, the findings from the survey were coded and analyzed using the Software Package for Social Sciences (SPSS version 14.0). The description of the finding was based on the descriptive analysis. The descriptive statistics was used to depict the attributes of the collected data, verify any violation of the principle assumptions of the statistical methods, and address the particular research questions (Pallant, 2007). Among the descriptive statistics applied include central tendency, and variation statistics such as mean, percentage, and standard deviation.

This study provides empirical proof of the research problem in Jordan in terms of the current practices and challenges of selecting the COTS software. The main deliverables and related details of this study are illustrated in Figure 3.3.

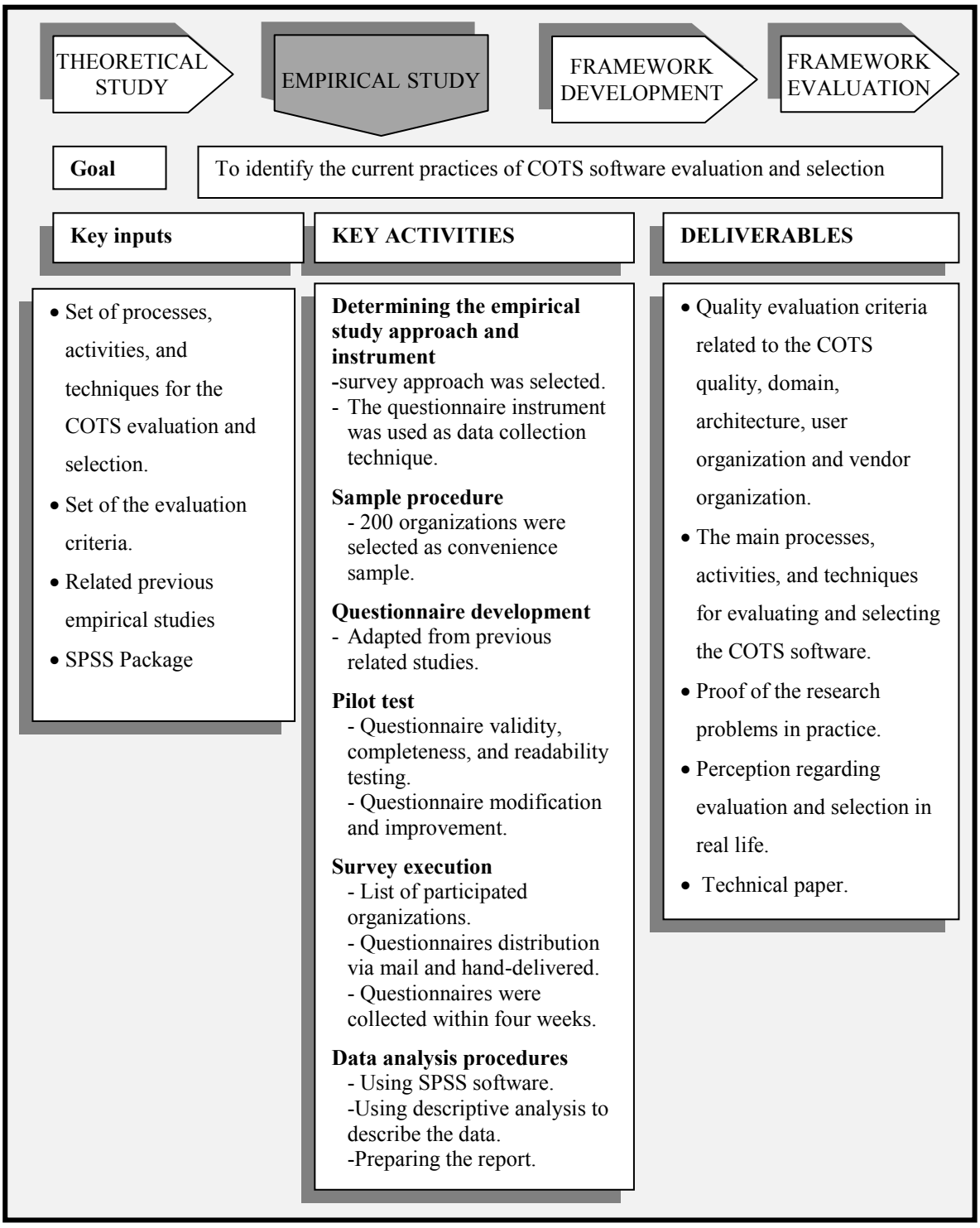


Figure 3.3. Empirical Study

3.5 Phase Three: Framework Development

This phase aims to design and construct a new framework for the COTS software evaluation and selection. The framework comprises of a set of concepts, ideas,

theories, principles and components that aims to support judgments and decision-making. It represents a conceptual structure intends to solve the research problem (Alvaro et al., 2010). In this phase, the new framework was developed based on the evaluation theory according to the following activities:

3.5.1 Identifying the Main Components of the Framework

The framework was constructed based on the three main elements: i) the six evaluation components provided by the evaluation theory, ii) the findings from the previous theoretical study such as the COTS mismatches (concepts, kinds, etc.) and techniques (GA and AHP techniques), iii) the findings of empirical study that provides the important evaluation criteria, techniques, processes in practice, as shown in Figure 3.4. In addition, the OTSO, PORE, and STACE methods were used as a starting point for identifying the components of this framework.

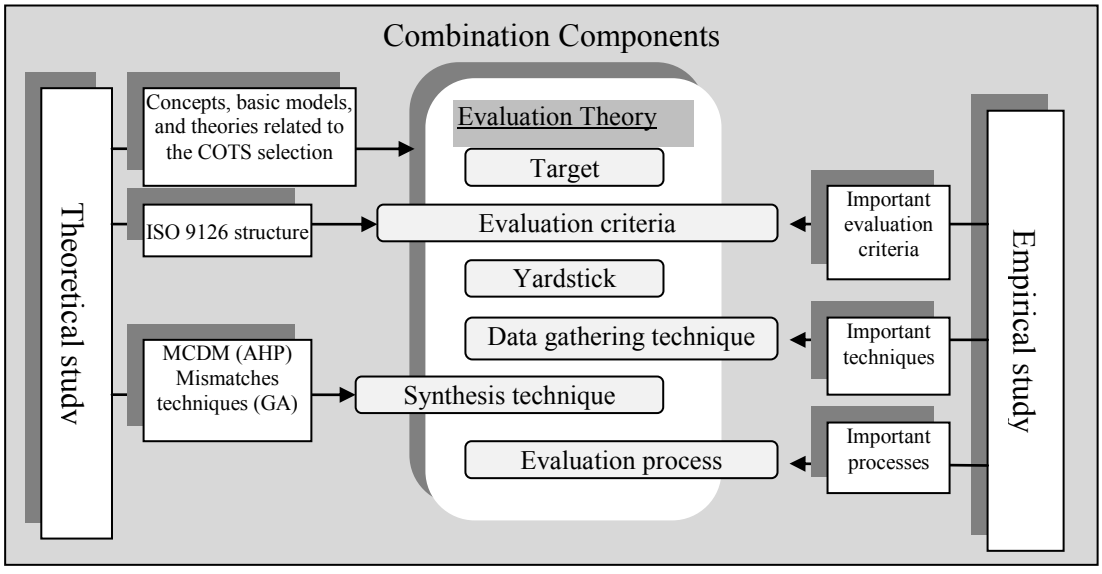


Figure 3.4. The Components of Combination

3.5.2 Developing the COTS Evaluation Criteria

In this research, the non-functional requirements problem was addressed in order to propose the important evaluation criteria for the COTS software. The COTS evaluation criteria were identified based on the findings of the theoretical and empirical study as shown in Figure 3.4. Since the development of a model can be time consuming and cumbersome Carvallo et al. (2004), the COTS evaluation criteria were constructed as a model according to the well-defined process or methodology presented by Franch and Carvallo (2003) (more details are in section 5.3.2). The hierarchy structure of the ISO 9126 model was used to decompose and present the evaluation criteria in a full hierarchy structure (characteristics, sub-characteristics, and attributes), as shown in Figure 3.5.

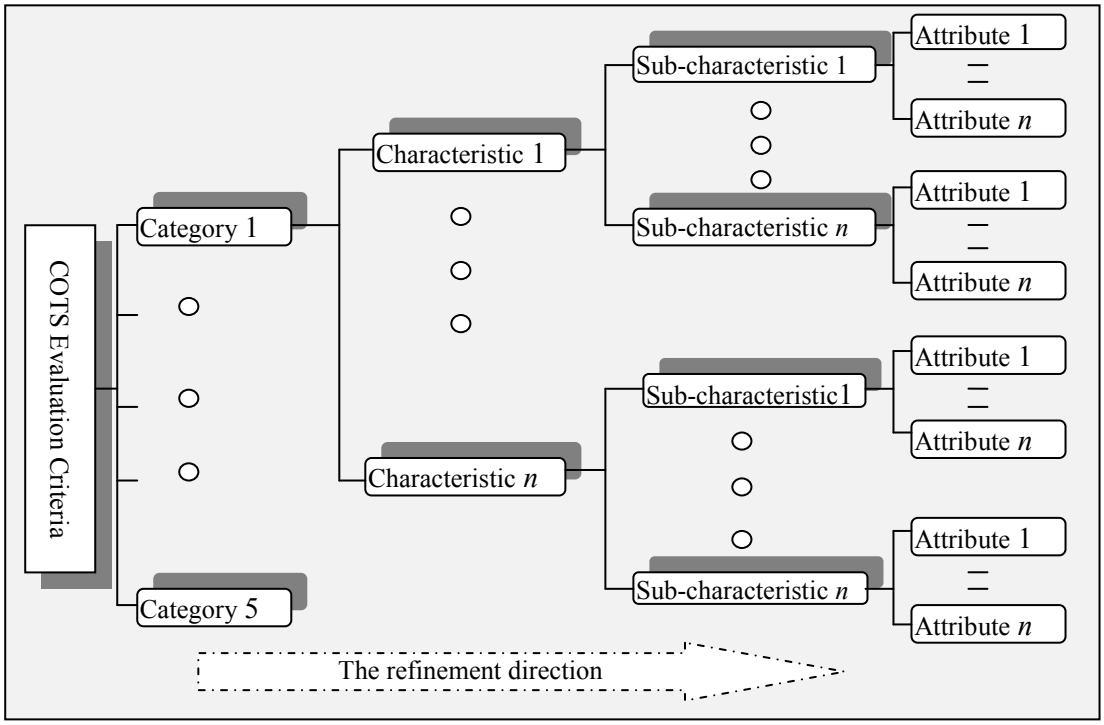


Figure 3.5. The Structure of Proposed Evaluation Criteria

3.5.3 Developing the Decision Making Technique

The other contribution of this research is a new decision making technique. This is a synthesis technique that aims to consolidate data and make appropriate decision. The decision-making technique was constructed based on the MCDM structure (hierarchy structure) because this is the most suitable for handling the multi-criteria decision problem by adapting the AHP and GA techniques. The combination of the new decision-making structure is shown in Figure 3.6.

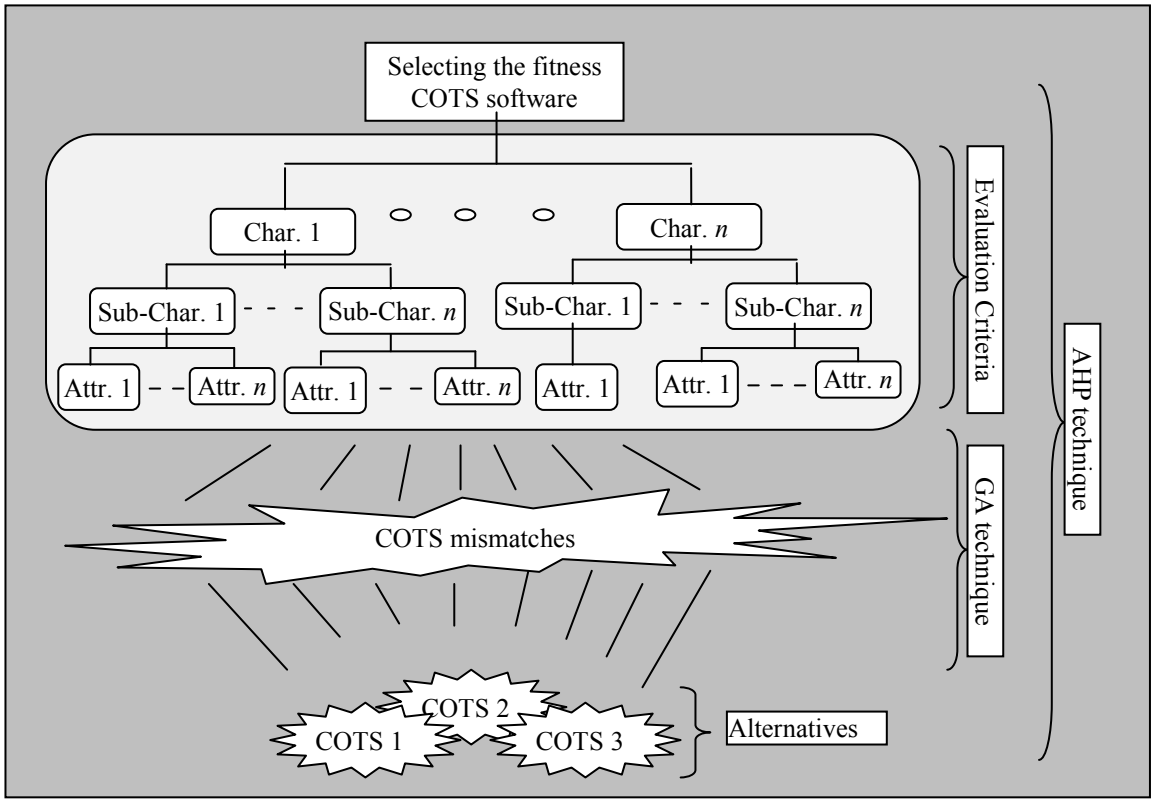


Figure 3.6. The Structure of Proposed Decision Making Technique.

3.5.4 Determining the Evaluation Processes

The evaluation processes play important role to carry out the evaluation. The evaluation processes consists of managing, linking, and performing various components of tasks to achieve the evaluation target. These tasks are determined from the previous empirical findings (Figure 3.4). The structure of these tasks is influenced by the structure of OTSO, PORE, and STACK selection methods.

3.5.5 Determining the Data Collection Technique

This component was adapted from the evaluation theory. Several data collection techniques were identified based on the COTS software information sources such as vendor and experimental group of COTS users. In addition, the findings from the empirical study have a role to determine the proper techniques. The techniques used are document review, joint application development design (JAD), evaluation form, and COTS software demonstration participation. More details are described in section 5.3.4.

3.5.6 Defining the Evaluation Target and Yardstick Components

The new framework also includes the delimitation of the evaluation target and yardstick components. The inclusion is based on the findings of other theoretical studies that applied the evaluation theory. The evaluation target is defined to determine the scope of the evaluation process. The yardstick component, which was constructed based on the work presented by Alves et al. (2005) and Mohamed (2007), stresses on the importance of defining the thresholds or acceptance values in

identifying the COTS mismatches. The yardstick or standard is represented by defining the ideal and lowest values for each attribute in the evaluation criteria in order to help determining the level and type (fully or partially mismatch) of mismatches. Detail descriptions are in section 5.3.3.

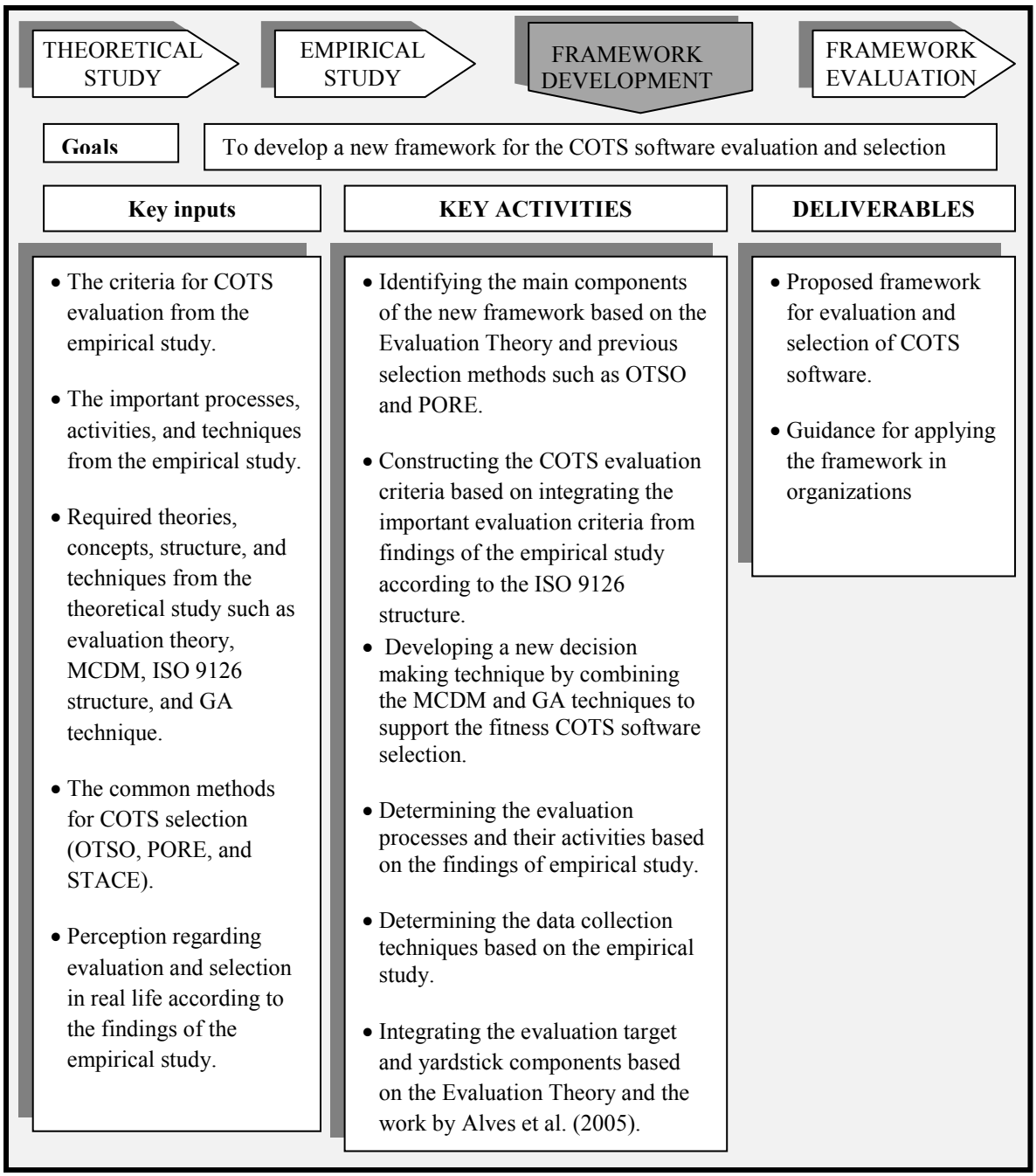


Figure 3.7. Framework Development Phase

3.6 Phase Four: Framework Evaluation

The framework evaluation, an important evaluation development phase (Behkamal et al., 2009; Dustin et al., 2002), aims to evaluate the applicability of the proposed framework to support the COTS software evaluation and selection in real life. In this context, two stages were carried out, verification and validation:

3.6.1 Verification Stage

The verification process is used to verify the validity of the proposed framework components. This was performed using the expert review approach, which is cheaper, easier, and faster. The approach was conducted through the following steps:

1. Identifying the potential experts related to the COTS software evaluation. As well as the experts should be local and international to collect diverse perceptions, they should also reflect the practice and theoretical points of view by including developers with at least 3 years experience in software development and procurement and researchers that hold an advanced degree (PhD.); faculty members at an accredited university; authorship; and have at least 5 years of experience (Hallowell & Gambatese, 2010; Rogers & Lopez, 2002).

2. Determining the technique or method for conducting the expert review approach. Delphi technique was used among experts review as the best technique to achieve the consensus between the experts (Moody, 2005). Delphi technique has become widely accepted method to achieve

convergence of perspectives regarding knowledge request from experts within specific domains. It was developed by Dalkey and Helmer (1963) to employs several rounds or iterations (feedback) designed to build consensus of experts' opinion (Hsu & Sandford, 2007).

3. The email and the interview approaches were used to contact with the experts. The email is used to contact with international experts when the interview will be so difficult to establish, while the interview approach can be used with the local experts where it easy to contact and conduct interviews with them.

The feedback from the identified experts were collected and analyzed to modify and improve the proposed framework. More details in section 6.2.

3.6.2 Validation Stage

After the proposed framework has been approved by the experts, it needs to be validated. Validation is the process of determining whether a model or framework is an accurate representation of the real world from the perspective of the intended usage (Thacker et al., 2006). Two approaches were used to carry out the validation stage: i) case study, and ii) yardstick validation.

The case study is chosen since the researcher cannot control or manipulate the relevant behavioural events. In addition, a case study is often suitable for research that is seeking to answer "how" and "why" questions (Yin, 2003). The case study implementation process involved the following activities:

1. Identifying the organizations that can participate in validating the framework. The potential organizations were selected based on their dealing with CBS, the available projects related to the COTS software selection, and its willingness to apply the framework. In this study, only two organizations, one in Jordan and Malaysia, agreed to apply the framework. This enables the testing of the feasibility and practicality of the framework in different environments and sittings.

2. Identifying the factors for estimating the proposed framework. The estimation factors have been identified according to Kunda (2002) and Kitchenham and Pickard (1998), as shown in Table 3.1.

Table 3.1

The Factors of Evaluating the Proposed Framework

(Adapted from Kunda, 2002; Kitchenham & Pickard, 1998)

Evaluation criteria	Variables
Gain satisfaction	- Perceived usefulness - Decision support satisfaction - Comparing with current method - Cost-effectiveness - Clarity - Appropriateness for task
Interface satisfaction	- Perceived ease of use - Internally consistent - Organization (Well organized) - Appropriate for audience - Presentation (readable and useful format)
Task support satisfaction	- Ability to produce expected results - Ability to produce usable results - Completeness - Ease to implementation - Understandability (easy to understand)

3. Data collection through interviews and document analysis. The interview method was selected because of its flexibility and adaptability in providing deeper understanding and useful information, helping to explore and understand complex issues (Sekaran, 2003). The interview was supplemented with documents analysis method that involved collecting, estimating, and analyzing the related documents to gain more information.

The COTS evaluation and selection framework is supported by the use of a prototype tool to select the fitness COTS software in a more systematic way. The prototype tool was developed using The Microsoft ASP.Net (Active Server Pages) technology with Visual Basic .Net (VB.Net) as the programming language. The prototyping development process consists of three components: i) user-interface, ii) computational module, and iii) database (See section 6.3.1.1).

The yardstick validation approach is used to validate the proposed framework by comparing it with ideal or baseline models in the same field. Using the yardstick beside others validation methods will increase the reliability of the validation process. In particular, if the model's components match with baseline models in the same field it may be taken as evidence that the model behaves correctly (Carson, 2002; Sargent, 2011). The yardstick validation starts by determining the ideal or the baseline models in the field of study. Then, the criteria for conducting the comparison are defined. Finally, comparing the proposed model with identified baseline models and discussing the results.

The outcomes of this stage are the confirmation of the validity of the proposed framework in practice. Figure 3.8 illustrates the framework evaluation phase.

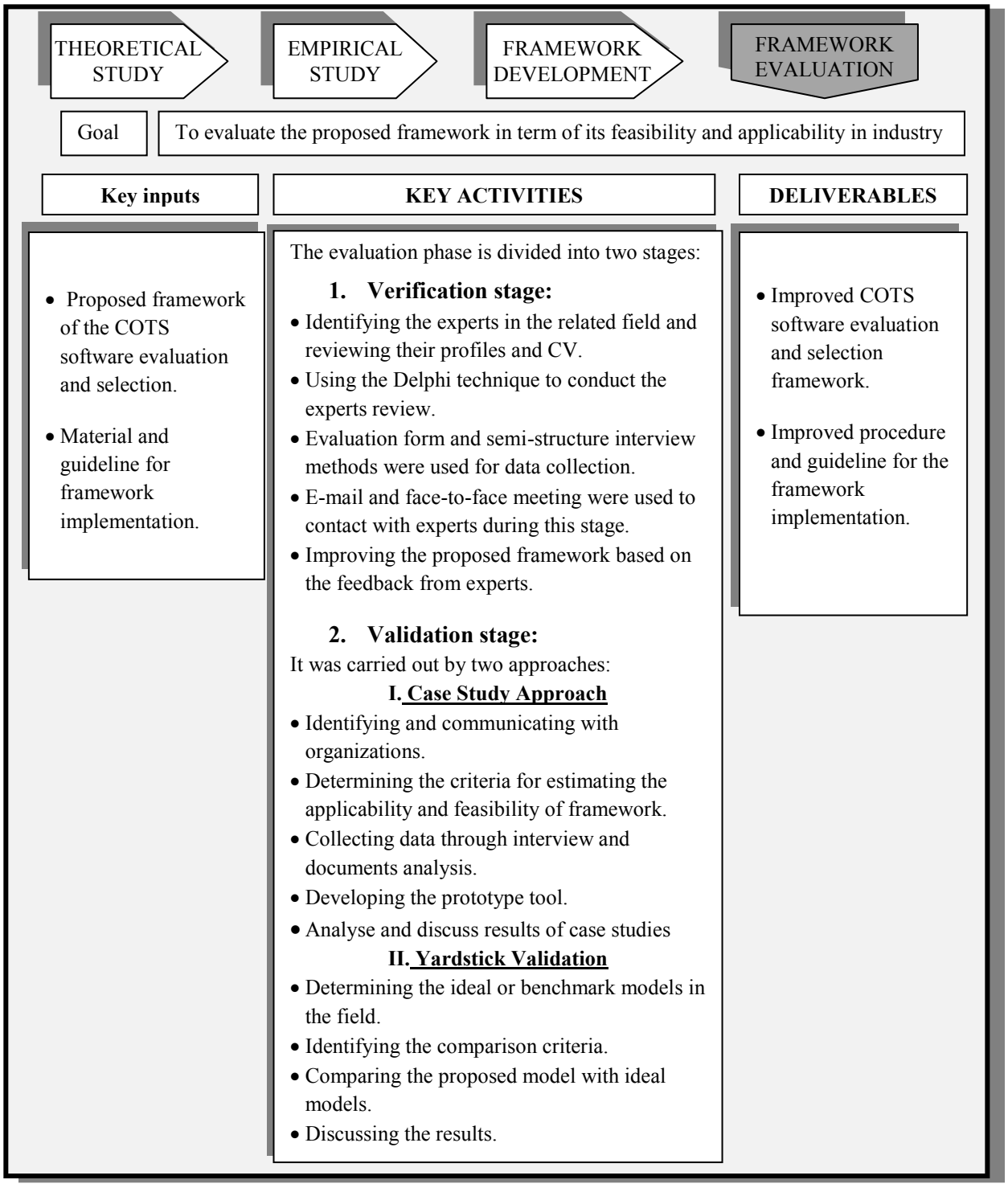


Figure 3.8. The Framework Evaluation Phase

3.7 Summary

The deduction approach was used as a research methodology in this study. Four phases were used to develop the new framework for evaluating and selecting COTS software: conducting theoretical study, conducting empirical study, developing framework, and evaluating framework. Each phase has key input, activities, and deliverables to achieve the research goal. Executing the entire phases guarantee the construction of a new framework for the COTS software evaluation and selection. The framework can support companies that are developing their CBSs and solve the research problem. The experts review approach was used to verify the identified evaluation criteria in term of their comprehensiveness, understandability, coherence, and accuracy. The case study and yardstick validation approaches were used to validate the proposed framework to ensure its feasibility and practicality.

CHAPTER FOUR
EMPIRICAL STUDY

4.1 Introduction

This chapter presents the findings of the empirical study conducted in Jordan. The study aims to investigate the practice of the CBD and COTS software evaluation and selection in Jordanian firms. It is also to understand the underlying issues of building systems from the COTS components. The findings from this empirical study would facilitate the development of the proposed COTS software evaluating and selecting framework and support the implementation of this framework to be acceptable in the real life.

The discussion in this chapter starts by explaining the questionnaire design. Next is the description on the data analysis, followed by the summary section.

4.2 Questionnaire Layout

The questionnaire was designed and established based on several studies such as the studies by Kunda (2002) and Gerea (2006) that consists of four main sections: i) demographic data; ii) CBS practice; iii) COTS evaluation and selection practice: and v) evaluation criteria. The sources of the identified variables in each section are explained in Appendix A. Full description of the questionnaire is in Appendix B.

4.2.1 Demographic Data

It is quite common to begin the questionnaire by gathering information related to the demographic data in order to identify and understand the respondents' profiles. This demographic section is divided into two parts: i) respondents' details including their job function in the organization; their experience with CBD; and their involvement in the current CBD activities; ii) organization details including its business function and number of employees. The questions in this section are in the form of check-box whereby the respondents can choose one or multiple answers. An example of each respective question is "*How long have you experienced building systems using the COTS components*", or "*What activities do you currently involved in?* (*Please check all that apply)"*.

4.2.2 CBS Practices

This section aims to elicit the current practices related to the CBD in the Jordanian organizations. Precisely, there are five questions relating to the COTS software, the current approach for integrating the COTS software, and the main risks and benefits of CBS. These questions are in the form of the five-point Likert scale and check-box questions. The five-point Likert scale format is based on Kunda's (2002) that used 1 to represent *strongly disagree*, *2* to represent *disagree*, 3 to represent *average*, 4 to represent *agree,* and 5 to represent *strongly agree*. An example of the check-box questions in this section is *"Please indicate the appropriate number of the COTS software products that you are using in your organization"*.

4.2.3 COTS Software Evaluation and Selection Practices

This section is included to investigate the best practices of the COTS software evaluation and selection. The main issues relating to the process of the COTS software evaluation and selection are also addressed. These comprise of its main problems, the current COTS software selection methods, its important processes and activities, the COTS mismatches consideration, and the supporting tools. Among the check-box and "Yes/No" questions posted are "*Please check the box(s) that describe the major problems that you face when evaluating and selecting the COTS software (Please check ALL possible choices)*", and *"Do you use any supporting tools during the COTS software evaluation and selection?"* respectively.

In addition, several techniques that have been identified in the theoretical study relating to defining the evaluation criteria; searching, data collection; identifying the COTS mismatches; and data analysis are investigated in this part of the questionnaire. These questions are presented using the five-point Likert scale as recommended by Kunda (2002), in which 1 represents *never*, 2 represents *rarely*, 3 represents *sometime*, 4 represents *regularly*, and 5 represents *always.*

4.2.4 Evaluation Criteria

The evaluation criteria section begins by examining the importance of the non-functional requirements in the COTS evaluation process. The level of importance is determined using the check-box question format. The main focus of this section is to identify the common and important evaluation criteria based on the list from the theoretical study, which is inherited from the non-functional requirements (technical

and non-technical). The criteria list is classified into five categories: quality; domain; architecture; user; and vendor organizations (Beus-Dukic, 2000) and measured according to the five-point Likert scale (1 represents very low consideration and 5 represents very high consideration).

4.3 Questionnaire Testing

Once constructed, the questionnaire underwent many rounds of review and revision. This was to ensure that not only the content is comprehensive and appropriate, but the layout should also be user friendly, the instructions should be clear, and the language should be understandable. These components were validated through a pilot test before distributing the questionnaires to the selected sample and collecting the actual data.

The pilot test was administered in thirteen organizations that are drawn from the population of interest in Jordan. The questionnaire was distributed face-to-face to various respondents who have extensive experience as developers, managers, and users. By conducting the pilot test, questions ambiguities, difficulties, incompleteness (to ensure all required items are included), and readability (to avoid misinterpretation of the posted questions) can be recognized. In addition, the time and motivation for answering the questions were also looked into. Having done the questionnaire testing, minor modifications over some of the questions were performed to improve the understandability and readability. Some of the unrequired questions were removed from the questionnaire, while two questions were relocated under a more suitable group. The required time and the respondent's capability to

complete the questionnaire can also be determined. The final questionnaire after the refinement is attached in Appendix B.

4.4 Data Collection and Response Rate

The main purpose of the data collection phase is to gather data from the representative sample. The questionnaire was the measurement instrument for the data collection, while the sample comprised of 200 Jordanian organizations. These organizations were determined based on the list given by the Jordanian Ministry of Industry and Trade and the Jordanian business website directory. These two sources were considered as the most up-to-date lists. The target respondent in the organization is the person who is responsible for evaluating and selecting the COTS software such as the developer, decision maker, or the manager. The respondents were given four weeks to fill up the questionnaire.

Table 4.1 indicates the total number of questionnaires that were distributed to the respondents and its response rate. The unreturned questionnaires are labeled as lost, while the incomplete questionnaires are considered as rejected. The rejected questionnaires are excluded from data analysis.

Based on Table 4.1, the response rate for this study is 31.5%. This denotes that the completed questionnaires are ready to be analyzed since Saunders et al. (2007) recommended that the reasonable average response rate is between the 30.0%-40.0%.

Table 4.1

Questionnaire Response Rate

Description	Organizations	Rate (%)
Sent	200	100.0%
Lost	110	55.0%
Received	90	45.0%
Usable	63	31.5%
Rejected	9	4.5%
No experience with COTS	18	9.0%

Next section describes the findings of the survey.

4.5 The Survey Findings

This section reports on the results of survey. The first section describes the demographic and background information on the participated organizations. The second section presents the current practice on CBS, while the third discusses the findings related to the COTS software evaluation and selection process. The final section addresses the identified evaluation criteria.

4.5.1 Demographic Data

Frequency distributions were used to categorize demographic data. The demographic data is presented in terms of the respondents' (job function, years of experience, and current activities in CBD) and organization backgrounds (primary business and number of employees).

4.5.1.1 Respondents Background

To understand their background, the respondents were asked to indicate their main job function, years of experience and current activities in CBD. Table 4.2 depicts that majority of the respondents are holding management post (28.6%) followed by systems analysis or design (15.9%), systems programming (15.9%), and academic or research (11.1%). The rests of them represent team leaders (7.9%), hardware specification (7.9%), and operations (6.3%).

Table 4.2

The Main Job Function in Organization

Job Function	Frequency	Percent
Management	18	28.6%
Systems analysis or design	10	15.9%
System programming	10	15.9%
Hardware specification	5	7.9%
Financial officer	1	1.6%
Operations	4	6.3%
Academic or researcher	7	11.1%
Team leader	5	7.9%
Other	3	4.8%
Total	63	100.0%

Table 4.3 portrays information relating to the respondents' experience working with CBD. Most of the respondents (73.0%) have less than 3 years working experience with CBD, while 19% have working experience between 3 to 10 years. Only a few of them (7.9%) has the longest involvement with CBD, which is between 11 to 20 years. This indicates that most of the employees that are responsible with the COTS software selection and integration in their organizations are actually lacking in experience in the CBD involvement.

Table 4.3

Work Experience with CBD

Experience	Frequency	Percent
Less than 3 years	46	73.0%
3 - 10 years	12	19.0%
11 - 20 year	5	7.9%
Total	63	100.0%

Regarding their current involvement in the CBD activities, most of the respondents acknowledged that they had been involved in more than one activity (Table 4.4). The CBD activity that has the most involvement from the respondents (36.5%) is the requirements engineering. This is followed by the COTS purchasing (31.7%), COTS identification (31.7%), COTS evaluation (30.2%), criteria definition (25.4%), COTS selection (23.8%), COTS integration (23.8%), COTS adaption (11.1%), and hardware specification (11.1%).

Table 4.4

Current CBD Activities

Activities	N	Percent of Cases
Requirements engineering	23	36.5%
Criteria definition	16	25.4%
COTS identification	20	31.7%
COTS evaluation	19	30.2%
COTS selection	15	23.8%
COTS purchasing	20	31.7%
COTS integration	15	23.8%
COTS adaptation	7	11.1%
Hardware specification	7	11.1%
Total	142	225.4%

**percentage of cases* is used to describe the data because it shows the percentage of the number of respondents who were chosen each item (it's appropriate for multi-responses question)

4.5.1.2 Organization Background

In terms of the organizational background, the respondents were asked to indicate the primary business and the number of employees in their organizations. Table 4.5 lists the primary business of the participated organizations. Education/training is found to be at the top of the list (25.4%). The ranking continues with manufacturing (12.7%), government agency (11.1%) and IT services (11.1%). Others include telecommunications, computer and system security, mining, service and public administration, finance, healthcare, insurance, and mobile and wireless technology.

Table 4.5

Primary Business of Participated Organizations

Business	Frequency	Percent
Bank/finance	2	3.2%
Insurance	2	3.2%
Manufacturing	8	12.7%
Computer/System Security	5	7.9%
Education/Training	16	25.4%
Mobile/Wireless Technology	1	1.6%
Telecommunications/network	6	9.5%
Mining	5	7.9%
IT Services	7	11.1%
Government	7	11.1%
HealthCare	2	3.2%
Service/Public Administration	2	3.2%
Total	63	100.0%

Table 4.6 shows the data distribution related to the number of employees in each participated organization. Most of the respondents come from large size organizations (46.1%), which comprise of a big number of employees of more than 250. 34.9% of participants are from small size organization of less than 51

employees, while 19.0% are from medium size organizations with 51 to 250 employees.

Table 4.6

Numbers of Employees in the Organization

The number of employee	Frequency	Percent
< 10	12	19.0%
10 - 50	10	15.9%
51 - 250	12	19.0%
> 250	29	46.1%
Total	63	100.0%

The subsequent section presents the findings of the descriptive statistic related to the CBS and COTS Software Selection.

4.5.2 Findings Related to CBS Practice

This section describes the survey findings related to the number of COTS software in the organization; the kinds of COTS software in the organization; the current approach for building CBS; and the benefits and risks of CBS. Basically, the respondents were asked to rate their agreement on those factors related to the CBS practice. The frequency and percentage were used as standard to describe and compare the relative importance of the identified variables.

4.5.2.1 Number of COTS Software in the Organization

Table 4.7

Number of COT Software in Organization

Number	Frequency	Percent
Less than 5	35	55.6%
5 - 10	14	22.2%
11 - 15	2	3.2%
16 - 20	9	14.3%
21 - 25	3	4.8%
Total	63	100.0%

Table 4.7 reveals that majority of the participated organizations (55.6%) used less than 5 COTS software products, while 22.2% of them used 5 to 10. Those that integrate a number between 16 to 20 COTS software into their systems represent 14.3%. Interestingly, only 4.8% of these organizations used more than 20 COTS software. This result suggests that most of these organizations prefer to build their own applications instead of procuring the COTS software.

Table 4.8 shows the connection between the number of the COTS products in the organization and the respondents' experience in dealing with CBD. The findings indicate that majority (69.6%) of the respondents, who only experienced dealing with CBD less than 3 years, work in the organizations that used less than 5 COTS software products. Those who have 3 to 10 years (41.7%) experience with CBD work in the organizations that have 5 to 10 COTS products, while those (80.0%) with 11 to 20 years of experience work in the organizations that have more than 11

COTS products. It can be concluded that majority of the organizations in Jordan has just started to use the COTS products. This also proves that the organizations need to leverage the benefits of COTS software even though without sufficient experience in dealing with CBD. In this case, it is necessary for these organizations to follow a well-defined and systematic method for selecting the appropriate COTS software to guarantee successful implementation of the final system.

Table 4.8

Number of COTS versus Work Experience

Experience with CBD (years)	Number of COTS					Total
	< 5	5 - 10	11 - 15	16 - 20	21 - 25	
less than 3 years	32 (69.6%)	8 (17.4%)	0 (0%)	5 (10.9%)	1 (2.2%)	46 (73%)
3 - 10 years	3 (25%)	5 (41.7%)	0 (0%)	2 (16.7%)	2 (16.7%)	12 (19%)
11 - 20 year	0 (0%)	1 (20%)	2 (40%)	2 (40%)	0 (0%)	5 (7.9%)
Total	35 (55.6%)	14 (22.2%)	2 (3.2%)	9 (14.3%)	3 (4.8%)	63 (100%)

4.5.2.2 Main Application of the COTS Software in the Organizations

Given the list of items related to the main application of the COTS software, the respondents were asked to select all applicable choices. Table 4.9 portrays the COTS software applications that are commonly used in the selected organizations. The most used applications are the database systems (73.0%), e-mail/messaging systems (55.6%), and accounting and finance (41.3%). Others include operating systems (38.1%), office automation system (38.1%), safety critical systems (25.4%) and business applications (19.0%). These organizations can be considered as inactive

users of the GUI builders, geographic information systems, and real time and embedded systems.

Table 4.9

The Common COTS Applications in Organizations

Applications	N	Percent of Cases
Database systems	46	73.0%
E-mail and messaging systems	35	55.6%
Office automation	24	38.1%
Real time and embedded systems	2	3.2%
Business applications	12	19.0%
Accounting and finance	26	41.3%
GUI builders	9	14.3%
Operating systems	24	38.1%
Safety criteria systems	16	25.4%
Geographic information systems	8	12.7%

4.5.2.3 The Current CBD Approaches

The literature pointed out that the three most popular approaches used by organizations in developing their systems are purchase and use/adopt; purchase and adapt; and purchase and integrate the COTS software (Kunda, 2002; Mohamed, 2007). The purchase and use/adopt approach refers to the manner in which the procured COTS software is immediately used without any adaptation or extension since it meets user's requirement. The purchase and adapt approach, on the other hand, is characterized by acquiring a single complete working system that satisfies most of the user's requirements but need to be adapted accordingly. The last approach, "purchase and integrate", means that purchasing a number of the COTS software, each satisfying part of user's requirements and integrating these

components into the system. In this survey, the respondents were asked to select from those three popular CBD approaches.

Table 4.10

CBD Approaches

Approaches	N	Percent of Cases
Purchase COTS and integrate	28	44.4%
Purchase COTS and adapting	35	55.6%
Purchase COTS and use	25	39.7%

Based on Table 4.10, the "purchase and adapt" approach was used by majority of the organizations (55.6%) in developing their CBS, while the "purchase and integrate" approach was only used by 28 organizations (44.4%). Only a few of these organizations (39.7%) choose to directly use the complete working COTS software system without adapting or extending. These results support the fact that the COTS software usually is not fully achieve the user's requirements which means that the mismatches problem between the COTS software and these requirements will raise in most cases. Thus, the appropriate decision making technique is needed to prevent selecting unfit COTS software that depletes the organization resources, such as time and budget, in the adaptation process.

4.5.2.4 Benefits and Risks of CBS

Using the Likert scale rating (1 represents the strongly disagree and 5 indicates to the strongly agree), the respondents were asked to assess their agreement to the items

related to the CBS benefits and risks. The frequency and percentage were used as standard to describe and compare the relative importance of the variables.

Table 4.11 indicates the agreement among the respondents regarding the main benefits offered by CBS. The number one benefit is the reduction of software development costs (79.4%), followed by the reduction of development effort (78.0%), an increase in system functionalities (60.3%).

Table 4.11

Benefits of CBS

Benefits	Strongly Disagree		Disagree		Don't Know		Agree		Strongly Agree	
	N	percent	N	percent	N	percent	N	percent	N	percent
Reduce development cost	2	3.2%	2	3.2%	9	14.3%	33	52.4%	17	27.0%
Reduce development effort	1	1.6%	6	9.5%	10	15.9%	31	49.2%	15	28.8%
Reduce maintenance cost	3	4.8%	4	6.3%	29	46.0%	23	36.5%	4	6.3%
Increasing COTS diversity	0	0.0%	6	9.5%	26	41.3%	31	49.2%	0	0.0%
Provides rich functionality	2	3.2%	1	1.6%	22	34.9%	29	46.0%	9	14.3%

On the other hand, Table 4.12 lists the various risks of building systems from COTS software as agreed by the respondents. Among the highest score are lack of vendor's support or after sales service (58.8%), and difficulties to select from the vast array of the COTS software (54.0%). The second risk supports the fact that there is a lack of well-defined process for selecting the fitness of the COTS software in industry.

Table 4.12

Risks of CBS

Risks	Strongly Disagree		Disagree		Don't Know		Agree		Strongly Agree	
	N	percent	N	percent	N	percent	N	percent	N	percent
Incompatibility with other components	3	4.8%	9	14.3%	29	46.0%	15	23.8%	7	11.1%
Periodic releases of COTS software	0	0.0%	17	27.0%	24	38.1%	22	34.9%	0	0.0%
Difficult to discover actual technical capabilities of COTS software	3	4.8%	16	25.4%	20	31.7%	21	33.3%	3	4.8%
Lack of support if COTS provider goes out of business	0	0.0%	9	14.3%	17	27.0%	26	41.3%	11	17.5%
Difficult to select from vast array of COTS software	0	0.0%	4	6.3%	25	39.7%	24	38.1%	10	15.9%
Additional functionality causes side effects	3	4.8%	5	7.9%	34	54.0%	19	30.2%	2	3.2%
Legal implications in case of system development failure and maintenance	1	1.6%	6	9.5%	27	42.9%	22	34.9%	7	11.1%

The following section describes the findings related to the COTS software evaluation and selection process.

4.5.3 COTS Software Evaluation and Selection

This section addresses the practice of evaluating and selecting the COTS software by describing the related problems and challenges; the current methods (i.e. formal or ad-hoc manner); and the processes and techniques of selecting the COTS software.

4.5.3.1 The Main Problems

Table 4.13 shows that the main problems encountered during the COTS software selection are lack of formal or well-defined process (79.4%), mismatches between COTS software features and user requirements (66.7%), failure of handling the non-functional requirements (63.5%), and difficulties to learn from previous related cases in their organizations (52.4%).

Table 4.13

Problems of the COTS Software Evaluation and Selection

Main Problems	N	Percent
Lack of formal process for evaluating and selecting COTS software	50	79.4%
Lack of considering COTS mismatches	42	66.7%
Lack of handling non-functional requirements	40	63.5%
Lack of learning from past COTS selection	33	52.4%

The relations between the lack of formal process for COTS software evaluation and selection and other problems are illustrated in Table 4.14. Several consequences of not following a formal or well-defined process as pointed out by the 50 respondents (79.4%) (Table 4.13) include failure to focus into the COTS mismatches problems (68%), handle the non-functional requirements (NFRs) (62%), and learn from previous evaluation and selection cases (54%). This indicates that by failing to adhere to a formal process causes the emergence of the other unanticipated problems. Examples of such problems are the inability to provide suitable techniques and mechanism in dealing with the COTS mismatches and to emphasize on vital role of the non-functional requirements.

Table 4.14

The Relations between Lack of Formal and Other Problems

Problems	Lack of formal process for selecting COTS software	Total (all cases)
COTS mismatches problem	34 (68.0%)	42 (66.7%)
Lack of handling NFRs	31 (62.0%)	40 (63.5%)
Lack of learning from past selection cases	27 (54.0%)	33 (52.4%)

4.5.3.2 Current Selection Methods

Regarding the current methods of the COTS software evaluation and selection, most of the respondents (85.7%) have not used any (Table 4.15). Only a small number of them is currently using specific method such as PORE (3.2%), STACE (1.6%), and CSSP (4.8%) in their organizations.

Table 4.15

Methods for Selecting the COTS Software

Methods	N	Percent of Cases
STACE	1	1.6%
PORE	2	3.2%
CRE	5	7.9%
CSSP	3	4.8%
None	54	85.7%

By cross tabbing the current methods and the related problems of the COTS software evaluation and selection process (Table 4.16), the results indicate that for those (85.7%) who do not use any systematic or formal method have high chances of facing problems such as COTS mismatches (88.1%), lack of handling non-functional requirements (95%), and lack of learning from past COTS selection cases (87.9%).

Although the STACE, PORE, CRE, and CSSP methods were used by few of respondents these methods suffer from lack of considering the mismatches between COTS features and user requirements like CRE (4.8%) and CSSP (7.1%), lack of handling non-functional requirements like PORE (5.0%), and lack of learning from past selection cases like STACE (3.0%) and CRE (9.1%).

Table 4.16

Current Used Methods Cross the Main Problems

The Main Problems	Current Used Methods					Total
	STACE	PORE	CRE	CSSP	don't use any method	
Lack of formal process for evaluating and selecting COTS	1 (2.0%)	1 (2.0%)	4 (8.0%)	1 (2.0%)	44 (88.0%)	50 (79.9%)
Mismatches problem	0 (0.0%)	1 (2.4%)	2 (4.8%)	3 (7.1%)	37 (88.1%)	42 (66.7%)
Lack of handling NFR	0 (0.0%)	2 (5.0%)	2 (5.0%)	0 (0.0%)	38 (95.0%)	40 (63.5%)
Lack of learning from past COTS selection	1 (3.0%)	1 (3.0%)	3 (9.1%)	0 (0.0%)	29 (87.9%)	33 (52.4%)
Total	1 (1.6%)	2 (3.2%)	5 (7.9%)	3 (4.8%)	54 (85.7%)	

The previous scenario shows that instead of using any formal method, most of the respondents prefer to use ad-hoc manners in evaluating and selecting the COTS software. The examples of the ad-hoc manners, as depicted in Table 4.17, include development team experiences (81.3%), managers' experiences (41.7%), developers–vendor relationships (37.5%), and relying on intuition (12.5%).

Table 4.17

Ad-hoc Manner to Select COTS Software

Ad-hoc Manner	N	Percent
COTS selecting based on the experiences of developers team	39	81.3%
COTS selecting based on the experiences of manager	20	41.7%
COTS selecting based on the intuition	6	12.5%
COTS selecting based on the relationship with vendor	18	37.5%

4.5.3.3 Supporting Tools

Almost all respondents (92.1%) have never use any supporting tool during the COTS software evaluation and selection process, while the rests (7.9%) use specific tool such as Microsoft Access and Excel (Table 4.18).

Table 4.18

Supporting Tools

Tool	Frequency	Percent
Yes	5	7.9%
No	58	92.1%
Total	63	100.0%

4.5.3.4 The Main Processes and Activities

Based on the findings in Table 4.19, majority of the respondents agree that all of the listed processes and activities are required in the COTS software evaluation and selection. These processes and activities are searching (61.9%), selecting (61.9%), documenting (60.4%), evaluating (60.3%), screening (57.1%), defining the evaluation criteria (54.0%), and planning (53.9%).

Table 4.19

The COTS Software Evaluation and Selection Processes and Activities

Main processes/Activities	Strongly Disagree		Disagree		Don't Know		Agree		Strongly Agree	
	N	percent	N	percent	N	percent	N	percent	N	percent
Defining the evaluation criteria	2	3.2%	2	3.2%	25	39.7%	24	38.1%	10	15.9%
COTS searching	2	3.2%	2	3.2%	20	31.7%	26	41.3%	13	20.6%
COTS screening	2	3.2%	8	12.7%	17	27.0%	30	47.6%	6	9.5%
COTS evaluation	2	3.2%	3	4.8%	20	31.7%	30	47.6%	8	12.7%
COTS selecting	2	3.2%	5	7.9%	17	27.0%	28	44.4%	11	17.5%
Documentation	4	6.3%	7	11.1%	14	22.2%	27	42.9%	11	17.5%
Planning	2	3.2%	4	6.3%	23	36.5%	30	47.6%	4	6.3%

4.5.3.5 The Most Frequent Used Techniques

To discover the most frequent techniques used in evaluating and selecting the COTS software, the respondents were asked to rate their agreement according to the given factors. For this purpose, the five-likert scale was used where 1= never, 2= rarely, 3= sometime, 4= regularly, and 5= always. The frequency and percentage were used as standard to describe and compare the relative importance of the items.

1) **Defining the Evaluation Criteria Techniques**

Defining the evaluation criteria is the first activity in performing the actual evaluation of the COTS software. There are many techniques proposed in the literature for defining these criteria. See Table 4.20.

Table 4.20

Techniques for Defining the Evaluation Criteria

Techniques	Never		Rarely		Sometime		Regularly		Always	
	N	percent	N	percent	N	percent	N	percent	N	percent
Brainstorm meeting	1	1.6%	0	0.0%	24	38.1%	24	38.1%	14	22.2%
Observation	1	1.6%	3	4.8%	19	30.2%	34	54.0%	6	9.5%
Prototyping and user demonstrations	0	0.0%	4	6.35%	21	33.3%	30	47.6%	6	9.5%
Use of scenarios	4	6.3%	6	9.5%	31	49.2%	20	31.7%	2	3.2%
Rich pictures and roots definitions	0	0.0%	8	12.7%	23	36.5%	27	42.9%	5	7.9%
Decision trees and tables	4	6.3%	9	14.3%	28	44.4%	16	25.4%	6	9.5%
Documents review	0	0.0%	6	9.5%	16	25.4%	29	46.0%	12	19.0%
Data flow diagrams	4	6.3%	8	12.7%	25	39.7%	16	25.4%	10	15.9%
Use of matrices	4	6.3%	9	14.3%	36	57.1%	13	20.6%	1	1.6%

Table 4.20 lists the various techniques used for defining the evaluation criteria. From the list, 65% (46% regularly and 19% always used) of the respondents use the documents review, 63.5% use observation, and 60.3% use brainstorming. The remaining includes prototyping and user demonstrations (57.1%), and rich pictures and roots definitions (50.8%). The techniques that got less than 50% of respondents' attention are use of scenarios, decision tree and tables, and data flow diagrams.

2) **COTS Alternatives Searching Techniques**

The main purpose of identifying the potential COTS alternatives that meet the user requirements is to enable a more rigorous evaluation. Table 4.21 indicates several COTS alternatives searching techniques that are available.

Table 4.21

Techniques for Identifying COTS Software

Techniques	Never		Rarely		Sometime		Regularly		Always	
	N	percent	N	percent	N	percent	N	percent	N	percent
Customer prior & past experience	0	0.0%	2	3.2%	25	39.7%	27	42.9%	9	14.3%
COTS inventory	2	3.2%	3	4.8%	8	12.7%	42	66.7%	8	12.7%
Prototyping and user demonstrations	0	0.0%	7	11.1%	33	52.4%	18	28.6%	5	7.9%
Market research	0	0.0%	2	3.2%	12	19.0%	39	61.9%	10	15.9%
Software development fairs and shows	2	3.2%	6	9.5%	36	57.1%	14	22.2%	5	7.9%
Provider adverts and promotions	2	3.2%	15	23.8%	20	31.7%	20	31.7%	6	9.5%
Internet search	0	0.0%	1	1.6%	32	50.8%	25	39.7%	5	7.9%
Software magazines	3	4.8%	11	17.5%	33	52.4%	15	23.8%	1	1.6%

The findings show that the COTS software inventory searching is the most frequent technique (79.4%: 66.7% regularly and 12.7% always used) used by the respondents for searching the COTS software. The next preferred techniques are market research (77.8%) and customer prior and past experience (57.2%), while 47.6% of the respondents always and regularly use the internet search to identify the COTS software. The remaining techniques in the list like providers' adverts and promotions (41.2%) and prototyping and user demonstrations (36.5%) got less attention by the respondents to use for identifying the COTS software.

3) **COTS Software Evaluation Techniques**

There are several kinds of techniques proposed in the literature to facilitate handling the data (collecting and analyzing) during the COTS software evaluation. This section presents the results regarding the data collection, data analysis, and COTS mismatches techniques.

a. Data Collection Technique

Table 4.22 demonstrates that the documents analysis is the most frequent technique used by the respondents (69.9%: 42.9% of regularly and 27% always) for collecting data. This is followed by experimentation users group advice (65%), attending demonstration (57.2%).

Table 4.22

Data Collection Technique

Techniques	Never		Rarely		Sometime		Regularly		Always	
	N	percent	N	percent	N	percent	N	percent	N	percent
Documents analysis	0	0.0%	1	1.6%	18	28.6%	27	42.9%	17	27.0%
Experimentation users group advice	0	0.0%	7	11.1%	15	23.8%	29	46.0%	12	19.0%
COTS demonstrations attending	0	0.0%	4	6.3%	23	36.5%	34	54.0%	2	3.2%
Questionnaires	2	3.2%	2	3.2%	29	46.0%	21	33.3%	9	14.3%
Algorithms for benchmarks testing	6	9.5%	11	17.5%	23	36.5%	21	33.3%	2	3.2%
Checklists	2	3.2%	5	7.9%	35	55.6%	18	28.6%	3	4.8%
Templates	4	6.3%	12	19.0%	21	33.3%	23	36.5%	3	4.8%

b. The Analysis Techniques

With regard to the analysis techniques, Table 4.23 portrays that the COTS demonstration attending technique is the most preferred by 69.9% of the respondents for the data analysis purposes, followed by the customer experience (68.2%) and the extensive experimentation (58.8%). It is important to mention that in the literature, the most used techniques are the MCDM techniques such as the AHP and WSM because in these findings, the MCDM is not preferable. The reason is that the evaluators do not have sufficient experience and well-defined method to deal with

these techniques especially when they require many calculations like in AHP technique.

Table 4.23

Analysis Techniques

Techniques	Never		Rarely		Sometime		Regularly		Always	
	N	percent	N	percent	N	percent	N	percent	N	percent
COTS demonstration attending	0	0.0%	2	3.2%	17	27.0%	33	52.4%	11	17.5%
Customer experience	0	0.0%	2	3.2%	18	28.6%	31	49.2%	12	19.0%
NFR framework	5	7.9%	12	19.0%	35	55.6%	11	17.5%	0	0.0%
Cards sorting and laddering	4	6.3%	13	20.6%	34	54.0%	11	17.5%	1	1.6%
Feature analysis technique	2	3.2%	15	23.8%	27	42.9%	17	27.0%	2	3.2%
Extensive experimentation	0	0.0%	3	4.8%	23	36.5%	27	42.9%	10	15.9%
Outranking methods	0	0.0%	17	27.0%	33	52.4%	12	19.0%	1	1.6%
AHP	3	4.8%	10	15.9%	30	47.6%	19	30.2%	1	1.6%
WSM	2	3.2%	13	20.6%	39	61.9%	8	12.7%	1	1.6%

c. The Importance of the COTS Mismatches

The mismatches between the COTS software features and user requirements need to be taken into consideration because it helps decision makers in selecting appropriate COTS software. As for the Jordanian organizations, most of them (77.8%) do consider the COTS mismatches issue, while the rest do not (22.2%) (Table 4.24).

Table 4.24

COTS Mismatches Considerations

Mismatches Considerations	Frequency	Percent
Yes	49	77.8%
No	14	22.2%
Total	63	100.0%

When the respondents were asked to determine the level of importance of the COTS mismatches, the results in Table 4.25 shows that 61.2% of those who took into

consideration the COTS mismatches indicated that identifying the COTS mismatches with user requirements is very important, while 20.4% of them admitted somewhat important, and 16.3% were not sure. Among the 22.2% of the respondents who did not consider the COTS mismatches, 64.3% of them are not sure and only 14.3% agree as somewhat unimportant.

Table 4.25

Considerations and Importance of the COTS Mismatches

Mismatches consideration	Importance of COTS mismatches				Total
	Very important	Somewhat important	Not sure	Somewhat unimportant	
Yes	30 (61.2%)	10 (20.4%)	8 (16.3%)	1 (2.0%)	49 (100.0%)
No	1 (7.1%)	2 (14.3%)	9 (64.3%)	2 (14.3%)	14 (100.0%)
Total	31 (49.2%)	12 (19.0%)	17 (27.0%)	3 (4.8%)	63 (100.0%)

d. The COTS Mismatches Techniques

When respondents were asked to select all applicable appropriate techniques for handling the COTS mismatches, 42.9% indicates that they never use any method or techniques (Table 4.26). For those who apply specific technique do so either through the traditional negotiations between user and the COTS software provider (36.5%), fulfillment (23.8%) or the used of GA (22.2%).

Table 4.26
The COTS Mismatches Techniques

Techniques	N	Percent of Cases
GA	14	22.2%
Negotiation	23	36.5%
Fulfillment	15	23.8%
Nil	27	42.9%

The findings are consistent with those of Mohamed et al. (2008) and Kvale et al. (2005) that identifying the mismatches between the COTS software and customer requirements has an important role for supporting the decision making in the COTS software selection process. Similarly, Mohamed et al. (2008), Alves et al. (2005), and this study, demonstrate that there is a lack of efficient technique in handling the COTS mismatches.

4.5.4 Overview of the Evaluation Criteria

This section presents the results regarding the general information about the evaluation criteria such as the non-functional requirements, and the following characteristics: quality, domain, architectural, user organization and vendor organization.

4.5.4.1 The Important of the Non-Functional Requirements

The non-functional requirements play a vital role in providing the criteria for evaluating and selecting the appropriate COTS software. In this context, Table 4.27 indicates that 77.8% of the respondents do consider the importance of the non-functional requirements during the COTS software evaluation and selection while 22.2% do not considered.

Table 4.27
Considerations of the Non-Functional Requirements

NFRs	Frequency	Percent
Yes	49	77.8
No	14	22.2
Total	63	100.0

The findings in Table 4.28 show that 59.2% of the respondents who are concern with the non-functional requirements do believe that it is vital for performing the COTS software evaluation and selection, while 30.6% think that it is somewhat important. On the other hand, for those who did not consider the non-functional requirements, 71.4% indicate unsure and 21.4% indicate somewhat unimportant.

Table 4.28

The Importance of the Non-Functional Requirements

NFR consideration	NFR importance					Total
	very important	somewhat important	not sure	somewhat unimportant	unimportant	
Yes	29 (59.2%)	15 (30.6%)	4 (8.02%)	1 (2.0%)	0 (0.0%)	49 (77.8%)
No	0 (0.0%)	0 (0.0%)	10 (71.4%)	3 (21.4%)	1 (7.1%)	14 (22.2%)
Total	29 (46.0%)	15 (23.8%)	14 (22.2%)	4 (6.3%)	1 (1.6%)	63 (100.0%)

As mentioned earlier, the non-functional requirements are classified into five characteristics: 1) quality, 2) domain, 3) architectural, 4) user organization, and 5) vendor organization. In order to determine the level of considering those characteristics, the respondents were required to use the five-point Likert scale with 1 refers to not considered and 5 indicates very high consideration. The mean and standard deviation (S.D) were calculated to find the central tendencies and determine the spread of values, and then used as standard to compare the relative importance of the variables.

According to Birisci et al. (2009) and Ismail et al. (2011), the interval levels or ranges of consideration for the five-point Likert scale were defined according to the following formula:

$(n-1) / n$ ……(4.1)

Where n is the maximum number in the used scale, which is to equal 5

The interval size of the consideration level between one through five was calculated as 0.80. Table 4.29 lists the calculated interval levels.

Table 4.29

Intervals Scale of the Consideration Level

Interval Scale	Level of Consideration
1 - 1.80	Nil
1.81 - 2.61	Low
2.62 - 3.42	Average
3.43 - 4.23	High
4.24 - 5	Very High

4.5.4.2 Quality Characteristics

Quality characteristics are set of the COTS software properties that can be described, measured, and evaluated. The quality characteristics used in this study are identified from the previous studies (see Appendix C). The findings indicate that the main quality characteristics with high and very high consideration include reliability, functionality, efficiency, usability, maintainability, reusability, and testability. The other characteristics are considered as average and low (Table 4.30).

Table 4.30

The COTS Quality Characteristics

Characteristics	N	Min	Max	Mean	S.D	Consideration Level
Efficiency	63	1.0	5.0	4.22	.77135	***High***
Expandability	63	1.0	4.0	3.37	.67922	Average
Functionality	63	3.0	5.0	4.32	.69155	***Very High***
Intra-operability	63	2.0	5.0	3.30	.66320	Average
Maintainability	63	2.0	5.0	3.97	.71771	***High***
Reusability	63	2.0	5.0	3.76	.79746	***High***
Reliability	63	2.0	5.0	4.43	.75595	***Very High***
Safety	63	2.0	5.0	3.37	.86699	Average
Survivability	63	1.0	5.0	3.24	.73428	Average
Testability	63	1.0	5.0	3.71	.86934	***High***
Usability	63	1.0	5.0	4.08	.80925	***High***
Verifiability	63	1.0	5.0	3.33	.84242	Average

4.5.4.3 Domain Characteristics

Domain characteristics are set of non-functional proprieties that describe the quality of the COTS software, which relate to specific domain and often reflect the fundamentals of the application domain. Table 4.31 shows that security is considered by majority of the respondents with score of very high.

Table 4.31
Domain Characteristics

Characteristics	N	Min	Max	Mean	S.D	Consideration Level
Specific type of hardware	63	1.0	4.0	2.56	.81869	Low
Timing constraint	63	2.0	5.0	3.29	.70548	Average
Security	63	2.0	5.0	4.29	.70548	***Very High***
Researched	63	1.0	5.0	4.22	.70584	***High***
Tested	63	2.0	5.0	4.02	.72938	***High***
Maturity of COTS software	63	3.0	5.0	4.16	.65270	***High***
Business policies and rules	63	3.0	5.0	3.79	.59997	***High***

The characteristics with high score include researched (popularity of the COTS software), maturity, tested, and business policies and rules characteristics. The other two characteristics, timing constraint and hardware, get average and low considerations respectively.

4.5.4.4 Architectural Characteristics

This type of non-functional characteristics is defined as set of quality characteristics describe the COTS software when integrating and interacting with other components in the system. Table 4.32 shows that the portability and interoperability characteristics are highly considered by most of the respondents, while the other characteristics (evolvability, integrity, scalability, composability, and flexibility) are rated as average. The results are established by assuming and counting the mean score based on Table 4.43.

Table 4.32

Architectural Characteristics

Characteristics	N	Min	Max	Mean	S.D	Consideration Level
Integrity	63	1.0	5.0	3.37	1.24825	Average
Portability	63	2.0	5.0	4.02	.65972	***High***
Flexibility	63	1.0	5.0	3.32	.81963	Average
Evolvability	63	2.0	5.0	3.40	.87140	Average
Scalability	63	1.0	5.0	3.27	.91944	Average
Interoperability	63	2.0	5.0	3.60	.68485	***High***
Composability	63	2.0	5.0	3.38	.68223	Average

4.5.4.5 User Organization Characteristics

Before acquiring the COTS software, the users need to be aware of a set of related characteristics and constraints. Table 4.33 depicts the respondents' answers related to each of those characteristics. The business issues is the main organization characteristics with very high score, while the current hardware platform, type of legacy applications (e.g. database applications), existing software development environment, and staff expertise and culture score high. The mean values of those characteristics with high score were much closed to the very high score. The timescale for components integrations and long-term strategy for software development got average consideration, while the political factor is not considered by the respondent.

Table 4.33

User Organization Characteristics

Characteristics	N	Min	Max	Mean	S.D	Consideration Level
Current hardware platform characteristics	63	1.0	5.0	4.22	.68261	***High***
Existing Software development environment	63	2.0	5.00	4.05	.60718	***High***
Staff expertise and culture	63	2.0	5.0	3.75	.67126	***High***
Type of legacy applications	63	2.0	5.0	4.10	.73428	***High***
Timescale for components integrations	63	2.0	5.0	3.37	.65504	Average
Long-term strategy for software development	63	2.0	5.0	3.27	.65270	Average
Political factors	63	1.0	5.0	2.52	.91329	Low
Business issues	63	2.0	5.0	4.24	.64042	***Very High***

4.5.4.6 Vendor Organizations Characteristics

Since buying specific COTS software requires a long time of dealing with the COTS supplier or vendor, the users need to recognize the characteristics of those vendor organization to ensure the continuity support of the product. In this study, the respondents were asked to indicate the level of consideration related to each of the vendor characteristics. The results suggest that all the identified vendor characteristics are highly and very highly considered. This means that the vendor characteristics are very important proprieties to be considered during the selection of the appropriate COTS software (Table 4.34). The main vendor characteristics with very high score are vendor reputation and vendor stability, while those with high score include supportability, sustainability, contract practice, experience, popularity, and vendor's certification.

Table 4.34

Vendor Characteristics

Characteristics	N	Min	Max	Mean	S.D	Consideration Level
Vendor reputation	63	2.0	5.0	4.37	.54777	Very High
Vendor stability	63	2.0	5.0	4.22	.65855	Very High
Vendor supportability	63	2.0	5.0	4.21	.62627	High
Vendor experience	63	3.0	5.0	3.94	. 77993	High
Vendor's popularity	63	2.0	5.0	3.92	.60379	High
Contract practice	63	2.0	5.0	4.13	.72091	High
Vendor certification	63	2.0	5.0	3.79	.84546	High
Vendor's sustainability	63	2.0	5.0	4.13	.65972	High

4.6 Discussion of the Findings

This survey investigates several issues related to the COTS software evaluation and selection. These issues are carefully discussed according to the following survey's objectives:

Objective 1: To verify the current practices of CBD in Jordanian firms.

- The most frequent approaches for building the systems from the COTS components are "purchase and adapt", and "purchase and use" (Table 4.10). In these approaches a single complete working COTS software product that satisfies most of customers' requirements is used either with adaptation for local needs or without any adaptation. This suggests that selecting COTS software product that achieves most of the customer requirements is more desirable than selecting more than one COTS product to meet the customer requirements (Intermediate System) (refer to section 2.2). Therefore, the success of the final system depends on the success of selecting the appropriate COTS software.

Objective 2: To determine the benefits and risks of using COTS software.

- The survey indicates that majority of the organizations are using the COTS software to reduce the costs and effort of systems development and increase the system functionality (see Table 4.11). Using the COTS software for building systems is cheaper because the total costs are shared with many users. These findings are consistent with most of other studies such as (Fang et al., 2010; Gayen & Misra, 2009; Li, 2006; Kiv et al., 2010; Sharma et al., 2007; Yanes et al., 2012; Yang et al., 2005). Despite the benefits, most of the organizations are

also worry about getting continuous COTS support especially when the vendor goes out of business, selecting from a vast amounts of the COTS software products (Table 4.12). This because most of the organizations in Jordan have not followed any well-defined process in selecting appropriate COTS software.

Objective 3: To identify the main problems and challenges related to the COTS software evaluation and selection.

- The survey indicates that 85.7% of the Jordanian organizations do not use any specific or formal method when evaluating and selecting the COTS software (Table 4.15). This indicates that they are using ad-hoc manners in selecting the COTS software such as by relying on the experience of the development team or depending on the relationships with specific vendor (Table 4.17). As a result, the organizations are facing various problems, which are: lacking of a well-defined and systematic method; difficulty in identifying the mismatches between the COTS features and customer requirements; failure in handling the non-functional requirements; and failure in learning from past selection cases (Table 4.13). This result corresponds to other studies such as (Alves et al., 2005; Beus-Dukic, 2000; Javed et al., 2012; Jingyue et al., 2009; Kunda, 2003; Lin et al., 2007; Mohamed et al., 2008; Neubauer & Stummer, 2007; and Sarkar, 2012).

- Referring to the Tables 4.24, 4.25, and 2.26, the survey reveals that majority of the organizations failed to identify and handle the COTS mismatches issue even though they are aware of its importance. The failure occurs because of they do not use any well-defined method or technique. Only a few of them has attempted to

solve some of the COTS mismatches using the negotiation method. This shows that many industries are not concern with the COTS mismatches particularly when they are not equipped with a well-defined decision making technique for addressing such issue.

Objectives 4: To identify the important processes, activities and techniques for evaluating and selecting the COTS software.

- From the developers' perspectives, the related activities of the COTS software evaluation and selection practices comprise of the evaluation criteria definition, COTS searching, screening, evaluating and selecting, planning, and documentation (Table 4.19). Several techniques are applied in order to facilitate the execution of those activities. Among the frequent techniques used for defining the evaluation criteria are documents review, observation and brainstorming (Table 4.20). For the COTS searching, the respondents choose the inventory searching, market searching, and customers' prior knowledge & experience techniques (Table 4.21). As for the data collection and analysis, the techniques applied are documents review, user group advices experimentations, and attending COTS demonstrations (Table 4.22 and Table 4.23). However, the findings showed that the MCDM techniques that have highly attention by the researchers in literature like AHP technique were not used by the most of the organizations (Table 4.23). One of the reasons for not using the AHP technique is due to certain limitations, which requires some form of complex calculation.

Objective 5: To identify the importance of the non-functional requirements of the COTS software evaluation and selection.

- The non-functional requirements play an important role in evaluating and selecting the COTS software. This fact is supported by the findings of this survey, where most respondents (77.8%) considered the non-functional aspects as important criteria for evaluating the COTS software (Table 4.27). This is consistent with various studies conducted by Ayala et al. (2011), Land et al. (2008), Pande (2012), and Javed et al. (2012).

Objective 6: To identify the common evaluation criteria for the COTS software evaluation and selection.

- In the survey, the non-functional characteristics were presented in five groups: quality, domain, architectural, user organization, and vendor organization characteristics. The results show that the most common quality characteristics of the COTS software assessment in the Jordanian organizations are functionality, reliability, efficiency, maintainability, testability, reusability, and usability (Table 4.30). Regarding the domain characteristics, five characteristics are considered as the most frequently used; security, business polices and rules, maturity, test, and research (Table 4.31). The two architectural characteristics used by most of the respondents are portability and interoperability (Table 4.32). In term of the user organization, the characteristics include the current hardware platform, business issues (financial case), type of legacy application, software development environment, and expertise and culture of the staff (Table 4.33). Finally, all the

identified vendor characteristics get a high attention from most organizations in Jordan (Table 4.34).

In addition, there are many facts that can be generated about the current state of the organizations in Jordan regarding their COTS software evaluation and selection. Firstly, these organizations use the COTS software products for developing their systems in order to save the costs and efforts instead of performing in-house development. Secondly, they are not using any well-defined method for evaluating and selecting the appropriate COTS software and thirdly, most of these organizations do not have enough experience to deal adequately with CBD. These facts are reflected in the findings shown in Tables 4.3, 4.7, 4.8, 4.11, 4.12, 4.13, 4.14, and 4.18.

In conclusion, the COTS software evaluation and selection process can be considered as problematic in the Jordanian organizations. Thus there is a pressing need for a systematic method to help the organizations improve their COTS software evaluation and selection implementation.

4.7 Summary

This chapter describes the construction and testing of the questionnaire, data collection and analysis. The survey aims to elicit and synthesize the current practices of the COTS software evaluation and selection in terms of its activities, techniques, and evaluation criteria involving various organizations in Jordan. Besides highlighting the problems of the COTS software selection, the achievement of the

survey objectives is also presented in this chapter. The survey results provide better understanding of how CBS can support the Jordanian organizations in developing and implementing more effective information systems. Most importantly, these findings form the basis for constructing the COTS software evaluation and selection. The detail of the proposed framework is discussed in the next chapter.

CHAPTER FIVE
COTS SOFTWARE EVALUATION AND SELECTION FRAMEWORK (COTS-ESF)

5.1 Introduction

Findings from the theoretical and empirical studies which were conducted during this research highlight the main problems in the COTS software selection such as the COTS mismatches and non-functional requirements problems which eventually initiated the effort for a new research. In this chapter, the solution for the problems which were described in the previous chapters is proposed as an improvement framework for the COTS software evaluation and selection purposes. The proposed framework may not only overcome those problems identified in the previous studies but also provide an applicable and systematic method of the COTS software evaluation and selection for the industry. The chapter starts by describing the main features of the proposed framework and then presents the details of the framework components.

5.2 The Main Features of the Proposed COTS-ESF

The proposed COTS-ESF provides a set of supporting features that would make it more acceptable to the system developers. In addition, the COTS-ESF is somewhat unique compared to the other methods pertaining to the COTS software evaluation and selection. The descriptions of the features are as follows:

1. This framework is developed based on the theoretical (theoretical study) and practical perspectives (empirical study), whereby most of the important processes, activities, techniques, and criteria were identified from the literature. These were then investigated in the real life in order to integrate the most acceptable features for the COTS-ESF.
2. The main components of COTS-ESF are explicitly defined and developed according to the six basic components derived from the evaluation theory. This theory is considered as a base and standard theory for any evaluation process in various disciplines and fields.
3. This research has proposed the COTS Evaluation Criteria (CEC) which emphasizes on the importance of the non-functional requirements for the COTS software evaluation and selection process. The CEC is developed based on the findings from the empirical study and by adapting some of the features from the previous models such as the Q'Facto 12 and Bertoa quality models (refer to section 2.3.4.2).
4. In order to offer a more reliable evaluation data and accurate decision, the COTS-ESF includes a new decision making process based on the combination of the AHP and GA techniques. Both techniques have their own advantages in which the AHP is the best technique for assigning weights for multi-criteria decision making (refer to section 2.3.2.2); and GA is the most appropriate technique for addressing the COTS mismatches (refer to section 2.3.2.3).

5.3 COTS-ESF

The need of developing a new framework for the COTS software evaluation and selection arises from the lack of well-defined solutions to handle the mismatches issue and the failure of identifying and classifying the non-functional requirements. The new framework is constructed based on the evaluation theory components. Figure 5.1 illustrates the main components of the proposed COTS-ESF followed by their detail explanations.

The proposed COTS-ESF consists of six components, which are evaluation target, evaluation criteria, yardstick, data gathering techniques, synthesis technique, and evaluation processes. These components are described in details in the following sections.

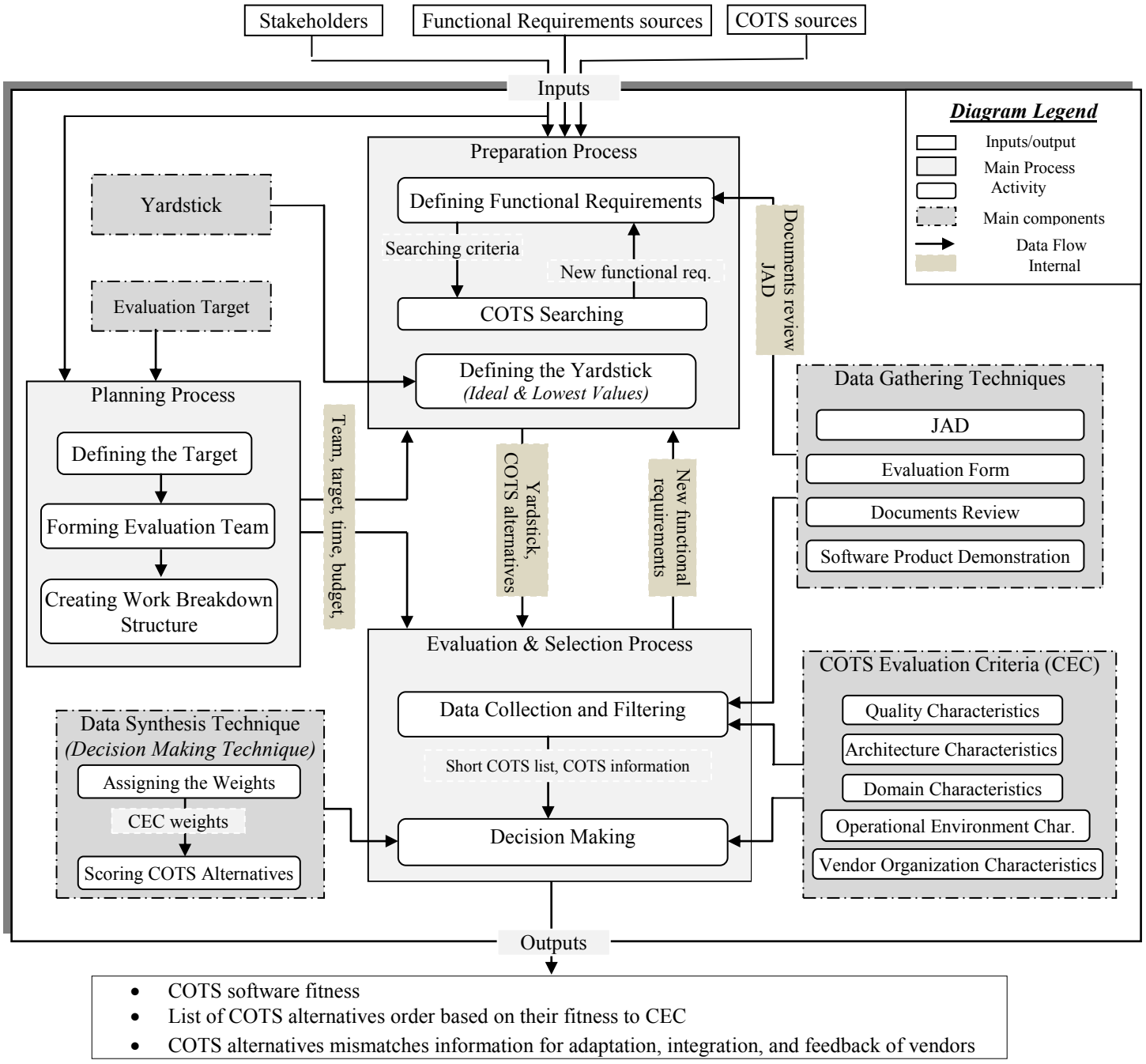


Figure 5.1. The Proposed COTS-ESF

5.3.1 Evaluation Target

It is necessary to define and delimitate the target under evaluation and describe target context. Defining the target helps evaluators to understand the kind of software that need to be analyzed during the evaluation. In this research, the general target of

developing a new framework is to evaluate and select the appropriate COTS software. This research starts by defining the target COTS software. There are many definitions found in the literature. However, most of the definitions are often very broad and covering variety kinds of products. This is the reason why both of researchers and practitioners are using the same term but referring to different meanings. In this thesis, the COTS software is defined based on the following characteristics: i) its existence is in a priori (ready-made) form developed by external party (vendor) for the purpose of gaining profit; ii) the product that can be bought, leased, and licensed; iii) it is available to the general public; iv) it is supported and evolved by vendor who retains the intellectual property rights; v) it is available in multiple and identical copies; and vi) it is used without source code modification (Morisio & Torchiano, 2002; Yanes et al., 2012).

5.3.2 Evaluation Criteria

In this component, the required criteria for evaluating the target are defined. Careful analysis of the non-functional requirements (NFRs) can support and improve the discrimination process between the competing COTS products that have already met the core functional requirements. Since defining the evaluation criteria is an essential and critical step in any evaluation process, especially in the COTS software evaluation, the COTS Evaluation Criteria (CEC) is constructed based on the review results of the state-of-theory and state-of-practice of the COTS software evaluation and selection (the results were discussed in chapter four). In addition, the criteria and their metrics have been identified and decomposed based on the analysis of a number

of related previous studies such as of those conducted by Bertoa and Valaillo (2002) and Kalaimagal and Srinivasan (2010/a). More detail about the references of the CEC criteria is included in Appendix C.

The CEC proposes new criteria related to vendor and user organization which are not provided by previous models. According to Beus-Dukic (2000) and Kaur and Mann (2010), the COTS software evaluation process requires to consider several kinds of non-functional requirements such as the requirements related to the system architecture, COTS domain, organizations (vendor organization and user organization). Similarly, five categories are established to classify the evaluation criteria, as shown in Figure 5.2, which are: quality, domain, architectural, operational environment, and vendor categories.

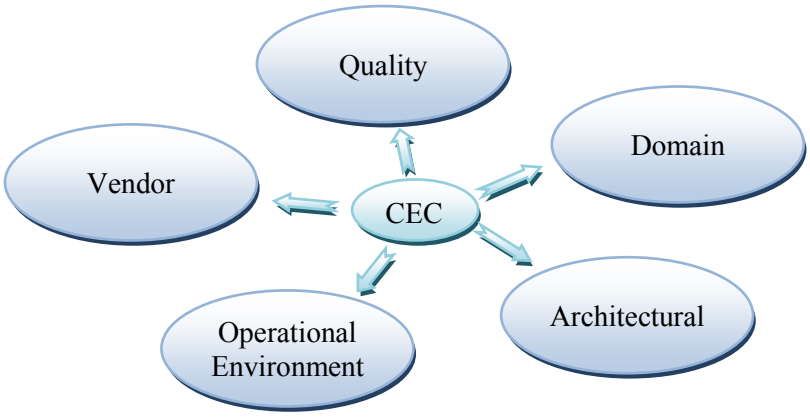


Figure 5.2. The CEC Categories

The CEC basically consists of four elements: i) categories, ii) characteristics, iii) sub-characteristics, and v) attributes (refer to section 3.5.2). The categories describe the related characteristics of particular part of the COTS evaluation, while the characteristic can be decomposed into several sub-characteristics. An attribute is a

measurable property of an entity. To make the CEC more accurate and applicable, several kinds of metrics are used to measure the attributes. These metrics were identified based on many studies such as those conducted by Alvaro et al. (2006), Bertoa and Valaillo (2002), and Kalaimagal and Srinivasan (2010/a). Table 5.1 shows the types of these metrics.

Table 5.1

The Types of Metrics to Measure the CEC Attributes

Metric	Description	Example of Attributes
Presence	This metric consists of Boolean value (1/0) to identify whether an attribute is present in the COTS software (means 1) or not (means 0).	Self-test, employee certification, error trace
Level	It is a subjectively measurement that is used to indicate the degree of effort, risk, ability, etc. using the five-likert scale that take several kinds of values such as (very high, high, etc.) or (excellent, very good, good, etc.).	Installation complexity, accessibility, replacement ease level
Percentage (Ratio)	This metric is used to describe the percentages values. It is represented by integer value between 0 and 100.	functionality coverage, excess, demonstration coverage
Value Integer (VI)	This metric is used to measure the exact value of the attribute under evaluation.	Required interfaces, COTS price, versions numbers
Time	It is used to measure time intervals using several kinds of time such as days, weeks, and years.	Time to use, time to configure, versions times

The CEC are in the form of a hierarchy structure consisting of four levels criteria. The first level contains the five evaluation criteria categories (quality, domain, architecture, operational environment and vendor), the second level includes 21 characteristics, the third level is the 55 sub-characteristics, and the fourth level consists of 119 attributes of the sub-characteristics measurement. The CEC categories, characteristics, and sub-characteristics are defined in Appendix D, while the metrics of each attributes are described in Appendix E.

The following sections discuss the CEC categories, characteristics, sub-characteristics and attributes.

a) **Quality Category**

The quality attributes refer to a set of characteristics and sub-characteristics that are used to describe and evaluate the quality of a software product. From the empirical study findings (refer to section 4.5.4.2), the main quality characteristics that received high and very high considerations are functionality, reliability, efficiency, maintainability, usability, and testability. Table 5.2 lists these characteristics along with their definitions.

Table 5.2

The Quality Characteristics

Characteristics	Definitions
Functionality	A set of capabilities, services, and functions that of COTS software in achieving the stated and implied needs when the software is used under specific conditions.
Reliability	The extent to which the COTS software is expected to perform its intended function with required precision.
Efficiency	The capability of the COTS software to provide appropriate performance, relative to the amount of resources used under certain condition.
Maintainability	The ability of COTS software to be modified. Modifications include corrections, improvements or adaptations to the software, due to changes in the environment, in the requirements, or in the functional specifications.
Usability	The effort required to learn, operate, prepare input, and interpret output of the COTS software.
Testability	The ability of the COTS software to provide some sort of tests or test suites that can be performed to the software to check its functionality inside (or in isolation of) the final system in which the software will be integrated.

Each of these characteristics has been decomposed into a set of attributes which are associated with different kinds of metrics as shown in the Table 5.3.

Table 5.3

Quality Category Decomposed Criteria

Characteristics	**Sub-characteristics**	**Attributes**	**Type**
Functionality	Suitability	Coverage	Ratio
		Excess	Ratio
		Service implementation coverage	Ratio
	Correctness	Precision	Ratio
		Computational Accuracy	Ratio
Reliability	Recoverability	Serializable	Presence
		Persistence	Presence
		Error handling	Presence
		Transactional	Presence
	Fault tolerant mechanism	Failure avoidance	Level
		Breakdown avoidance	Level
		Incorrect operation avoidance	Level
		Incorrect operation mitigation	Level
Efficiency	Resource behavior	Memory utilization	Integer
		Disk utilization	Integer
	Time behavior	Response time	Level
		Throughput	Level
		Capacity	Level
Maintainability	Changeability	Customizability	Integer
		Customizability Ratio	Integer
		Change Control Capability	Level
	Ease of migration	Migration ease level	Level
	Stability	Stability level	Level
Usability	Learnability	Time to use	Time
		Time to configure	Time
		Time to admin	Time
		Time to expertise	Time
	Understandability	The quality of help system	Level
		Computer documentation	Presence
		Existing Training course	Presence
		Demonstration coverage	Ratio
		Quality of user document	Level
	Operability	Provide interfaces	Integer
		Required interfaces	Integer
		Effort for operating	Level
		Tailorability	Level
		Administrability	Level
Testability	Test document	Test suit document	Presence
		Proofs of previous tests	Presence
	Start-up self-test	Self-test	Presence
		Environment test	Presence
	Traceability	Performance trace	Presence
		Error trace	Presence

b) Domain Category

The COTS domain category is defined as a set of common properties that reflect the extent of the COTS software success in specific domain such as military, security or safety, and financial. The domain quality characteristics are required because the COTS software developers have no experience in specific domain and no direct contact with end-users. This causes the end-users to be left alone with a difficult task to evaluate and select a specific COTS software basis on domain-specific requirement.

The following are the most important characteristics as determined in the empirical study (refer to section 4.5.4.3): security, maturity, and popularity. Table 5.4 lists these characteristics with their definitions.

Table 5.4

Domain Characteristics

Characteristics	**Definitions**
Security	Ability of COTS software to prevent unauthorized access, whether accidentally or deliberately.
Maturity	The ability of the COTS software to meet the needs of reliability under normal operation, which is represented by the number of commercial versions that have been marketed, and the interval between those versions.
Popularity	The level of acceptance or usage among various users of certain domain (The more popular software is more reliable).

These domain characteristics are then decomposed into set sub-characteristics and attributes as depicted in Table 5.5.

Table 5.5

Decomposed Criteria of Domain Category

Characteristics	Sub-characteristics	Attributes	Type
Security	Data protection	Data encryption	Presence
		Preventing data corruption	presence
	Controllability	Execution control	Presence
		Environment control	Presence
		Function features control	Presence
	Auditability	User access recording	Presence
Maturity	Volatility	Versions times	Time
	Evolvability	Versions numbers	Integer
	Failure removal	Bugs fixed	Integer
Popularity	Number of users	Installations/setup	Integer
		Upgrades	Integer
	Locatability	Accessibility	Level
	Internet discussions	Views of information page	Integer

c) **Architectural Category**

Some set of the non-functional requirements should describe the software component integration and their interactions. Among the architectural characteristic measures include reusability, portability, interoperability, safety in use, and operation supportability. Table 5.6 shows these characteristics with their definitions.

Table 5.6

Architectural Characteristics

Characteristics	Definitions
Reusability	The ability to reuse existing COTS software to be integrated in the system under development or create a more complex system.
Portability	The ability of the COTS software to be transferred from one environment to another.
Interoperability	The ability of COTS software to interact with other systems whether they are composition of components or not.
Safety in Use	The level of the COTS software safety when used by end users on the system.
Operation support	The ability of the COTS software to provide helpful information for identifying and resolving issues when it fails to work correctly.

Each of these architectural characteristics is decomposed into sub–characteristics and attributes as shown in Table 5.7.

Table 5.7

Decomposed Criteria of the Architectural Category

Characteristics	**Sub-characteristics**	**Attributes**	**Type**
Reusability	Generality	Domain abstraction	Presence
		History of reuse	Presence
	Hardware/software independency	Hardware dependency	Presence
		Software dependency	Presence
Portability	Installability	Installation Document	Presence
		Installation complexity	Level
	Deployability	Deployment document	Presence
		Deployment complexity	Level
	Adaptability	Mobility	Presence
	Replaceability	Replacement ease level	Level
Interoperability	Compatibility	Data compatibility	Presence
		Version compatibility	Presence
Safety in Use	Risk of software	The risk level	Level
Operation support	Diagnostic information	Monitoring system (*activity &performance*)	Presence
	Troubleshooting	Snapshot system's state	Presence
		Detailed operational and functional reports	Presence
		Logging and auditing information	Presence

d) **Operational Environment Category**

There are a variety of characteristics related to the implementation environment (user organization) that should be considered when evaluating and selecting the COTS software. Findings of the empirical study indicate that the most common and important of those characteristics are: system platform, software development environment, culture, and financial issue. All of these characteristics are defined in Table 5.8.

Table 5.8

Operational Environment Characteristics

Characteristics	Definitions
System Platform	A combination of hardware and software platforms. A hardware platform is a family of architectures used to launch software, while the software platform is for wrapping the essential parts of hardware platform.
Software Development Environment	Contains everything required by the developers' team to adapt and integrate the COTS software into a final system.
Culture	Represents the current skills, knowledge, and expertise of software users. It also represents the business roles, language, business symbols, and behavior in the organization.
Financial Issue	The financial ability of the organization.

The operational environment characteristics are then decomposed into several sub-characteristics and attributes with associated metrics as shown in Table 5.9.

Table 5.9

Decomposed Criteria of the Operational Environment Category

Characteristics	Sub-characteristics	Attributes	Type
System Platform	Hardware platform	Processing unit performance	Level
		Memory system	Level
		Data transfer system	Level
	Software platform	Current operating system	Presence
		Current middleware (e.g. CORBA standard)	Presence
		Communication applications	Presence
Software Development Environment	Process	Development process *(tasks, roles, processes)*	Level
		Supplementary process *(standards, guidelines)*	Level
	Technology	Development tools *(Integration and configurations tools)*	Level
	People (developers)	Developers' Skills/knowledge	Level
Culture	User culture	Expertise	Level
		Users' Knowledge/skills	Level
		Expectations	Level
	Organizational culture	Behavior *(General operating norms, Interaction)*	Level
		Symbols	Level
		Language	Level
		Policies and roles	Level
Financial Issue	Acquisition costs	COTS price	Integer
		Delivery (installation) cost	Integer
		Training cost	Integer
		Infrastructure upgrading cost	Integer
	Further development costs	Adapting cost	Integer
		COTS testing cost	Integer
		Integrating cost	Integer

e) **Vendor Category**

The organization that developed, maintained, upgraded, and supported chain of COTS products in the market has a significant influence in the selection process and a high chance of success in the final project development. However, among the anticipated risks that may are caused by vendor include inability to provide product documentation, incompatible product modifications, and inadequate long time support service. Thus, a set of the important characteristics are proposed based on the findings of the empirical study to check the quality of the vendor before selecting a product (Table 5.10).

Table 5.10

Vendor Characteristics

Characteristics	**Definitions**
Reputation	Users' view on vendors. It is important for a user to find out others' perceptions on a particular vendor before making any procurement decision.
Stability	An indication of having future support and upgrades from the vendor.
Supportability	The ability of the vendor to provide a set of services, maintenances, and upgrades during and after the software product implementation.

Each of these vendor characteristics is then decomposed into various sub–characteristics and attributes as shown in Table 5.11.

Table 5.11

Decomposed Criteria of the Vendor Category

Characteristics	Sub-characteristics	Attributes	Type
Reputation	Certification	Employee certification	Presence
		Development process certification	Presence
		Software product certification	Presence
	Reference checks	List of clients	Presence
	Market coverage	The number of Customers	Integer
	Competence	Flexibility of development process	Level
		Using last technology	Level
Stability	Financial	Financial ratio	Level
	Track record	Time in business	Time
		Time in development this software	Time
	Employees	Number of employees	Integer
	Strategy	Long-term strategy	Presence
Supportability	Delivery	On time delivery performance	Level
		Confirmation software functions	Level
	Quality of training	Quality of training courses	Level
		Training tool/technology	Level
	User support and communication	Help desk support	Level
		User queries/faults	Level
		Remote or online support	Level
	Software support	Releasing functional software upgrade	Level
		Software upgrade path	Level
		Services warranty support	Level

5.3.3 Yardstick

The yardstick component also known as standard consists of specifications, requirements, descriptions, or values for each evaluation criterion. The general structure of the yardstick is developed based on the proposed evaluation criteria. The main aim of this component is to provide the threshold values (ideal and lowest values) to each evaluation criterion. These values represent the standard for comparing the values of all COTS software in order to achieve positive evaluation. There are two types thresholds values defined in this component:

1. The ideal value that represents the target or ideal value to achieve by COTS software to each attribute in CEC.

2. The lowest value that represents the minimum or acceptance value to each attribute in the CEC. It also helps to determine whether the COTS software is will be accepted with certain mismatches or rejection.

The ideal and lowest values in the yardstick are determined by the evaluation team based on the user requirements. If possible, the values are represented in pair such as [criterion, datum/information]. This is depicted in Figure 5.3.

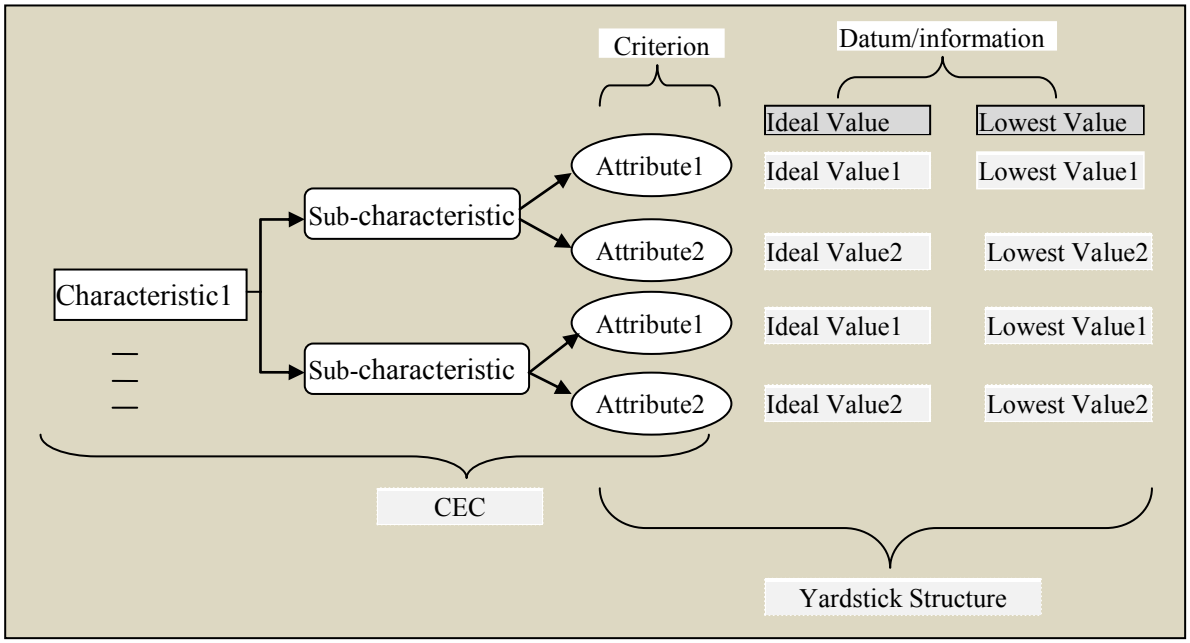


Figure 5.3. Yardstick Structure

The yardstick is an important component in addressing the following two points:

Firstly, it is very helpful in addressing the COTS mismatches at each attribute between the ideal value that provided by the evaluation team with the current value that provided by the COTS product at each attribute. See the example in Table 5.12.

Table 5.12

Example of Defining and Using Yardstick

Attribute	Ideal value	Lowest value	COTS value	Identifying mismatch
Functionality coverage	95%	80%	85%	Partially mismatching
			95%	No mismatching
			50%	Full mismatching

Secondly, defining the yardstick has a vital role in filtering the COTS software that provides values less than the lowest values of the attributes. In other word, the COTS software that fails to achieve the lowest value at each attribute will be rejected and eliminated from the COTS alternatives list. The values of the yardstick thresholds (ideal and lowest values) are determined by the evaluation team using the Joint Application Design (JAD) technique. This technique considers several perspectives such as domain expert, technical, end-user, and manager in defining accurate values.

It is worth to mention that there are different data types of criteria that should be considered when determining the ideal and lowest value. These types are explained in Table 5.13 (refer to Table 5.1).

Table 5.13

Data Types of Attributes in the Yardstick

#	Attributes	Type	Ideal value	Lowest value
1	Functionality coverage	Percentage (%)	90%	80%
2	Software price	Integer	1000$	1600$
3	Training availability	Presence (1/0)	1(Existing)	0 (Not existing)
4	Processing unit performance	Level	1(Very high)	3 (Average)
5	Time in business	Time	5 years	2 years

The yardstick template is found in Appendix E.

5.3.4 Data Gathering Techniques

Typically, different kind of criteria requires different kind of data. Therefore, one or more techniques are required to collect this data in order to judge and select the fittest COTS software. Precisely, in the COTS software domain, there are four data resources to provide the required information related to the CEC attributes of each COTS software alternative. These resources are:

i) The stakeholders in the user organization who have information related to the functional requirements; their expectations; and experiences. Such of those stakeholders are the end-users, developers, managers, and domain experts.

ii) The initial information in the available documents related to the COTS software and user organization such as the COTS price and hardware storage space.

iii) Experimental group of users who used the COTS software and they have experience with the COTS software domain and its' vendors.

iv) The COTS software vendors also represent other important resources of some required information about the COTS software such as COTS supportability.

v) The COTS software demonstration that provide required information during the running of the COTS software such as response time.

Each of these resources requires data gathering technique. Based on the findings of the empirical study (refer to section 4.5.3.5), there are several techniques used for collecting data from various data resources, which are: Joint Application Design (JAD), documents review, evaluation form, and the COTS software demonstration participation. Table 5.14 lists these techniques together with the data resources.

Table 5.14

Data Gathering Techniques Mapping With Data Resources

Resources	**Technique**
Information related to the Stakeholders	JAD
Vendor and user organizations documents	Documents review
Experimental users group	Evaluation form
Vendor	Evaluation form
COTS software	COTS software demonstration attending

1) ***The Joint Application Design (JAD)*** is a technique that allows the users group to work together to identify, develop, and manage the system requirements. The JAD technique is still the best technique for collecting user requirements through a series of meetings (brainstorming session). Thus, this technique is adopted in this research to collect the requirements, determine the evaluation criteria, and assign the criteria values. Meetings are organized to work together with the software users, managers, experts, and vendors. The main feature of this technique is that it can determine the criteria based on the available features or characteristics of the COTS software in the market by selecting the domain expert and vendor agency to join in the JAD team.

2) ***The documentation review*** is a basic and the most common technique in the software development process. This technique is performed by studying the documents that are provided by the organization. These documents consist of a set of articles, websites, and reports that discussed and provided many kinds of information about the design, characteristics, requirements, guidelines, and implementation process of specific system or software. In this context, two types of the documents are needed to review and analyze in supporting the data gathering process:

   i. The user organization document provides information about the operational environment such as those related to the existing system, infrastructure, users' requirements and others.

   ii. The COTS software document provides information about the COTS price, some of the COTS characteristics (i.e. reliability and functionality), usage and installation guideline, and others. This kind of document may also comprise of various information such as the COTS product website, the advertisements, the vendor's shows, or other related document.

   The team developers will review these documentations and extract the required information or values that can be assigned to the evaluation criteria.

3) ***The evaluation form*** is used to gather information especially when there are a large number of persons to obtain from. This technique is suitable to collect the data when the respondents are from different places or countries. The

evaluation form is used to collect the required data about the set of criteria from the COTS software vendors. In addition, the form will also be given to an experimental set of users to evaluate the same COTS software. The evaluation from is developed by forming set of questions based on the attributes related to specific data resource. Figure 5.4 shows an example of the evaluation form that used in collecting the data from the vendor. The full evaluation forms are shown in Appendix F.

Company Name:

Product Name:

Version :

Dear vendor,
Computer center in AL-Hussein Bin Talal University strives to seek out vendors offering the highest quality products. To be considered, please supply us with the information requested below about your product.

#	Questions	Answer
1	How much the number of customizable parameters that the software product offers?	
2	How much the number of parameters offered by the software product compared with the number of its provided interfaces?	

	Please, answer the following questions by Yes (Y) or No (N)	Yes	No
20	Does a proper documentation has been provided for plans, tests cases, test environment, etc.?		
21	Does the history of previous tests has been provided which demonstrate how successful or unsuccessful the tests were?		
22	Can the software product deal with encryption in order to protect the data it handles?		
23	Can the software product be reused across different domains related to the specific functionality that provided by the software product?		

Figure 5.4. Snapshot of the Evaluation Form

4) ***The COTS software demonstration Participation*** is a method for running and accessing trial version of the software product in virtual environment in order to gather the data about the COTS software and verify the information

from the vendor documents. This technique is conducted through two methods:

i. The vendors of the COTS software alternatives are invited to provide software demonstration in the presence of the team developers. This gives the evaluators an opportunity to interact fact-to-face with the vendors to get more information and understanding about the COTS capabilities. This technique is more appropriate for local vendors.

ii. The vendors offer the COTS software demonstrations through the manufacturers' websites that enable the evaluators to download the trial version of the COTS with limited functionality. By attempting to go through the trial version, the evaluators can have direct interaction with the software. This is more efficient especially when the vendors are out of town.

Even though the software demonstration gives the team developers great opportunity to identify and access the functionality and features of the software product before buying it, this technique requires a long time and a lot of effort in organizing and managing the meeting with vendors.

5.3.5 Synthesis Technique

Once the required data are collected, the synthesis techniques should be applied to synthesize all data and compare against the yardstick that contain the threshold

values of criteria in order to select the fittest COTS software. To do so, it is important to use an appropriate data synthesis technique since the final decision of selecting the COTS software will be based on the results of this technique. Basically, the new decision making technique is developed based on the combination of the AHP and GA techniques.

The main steps of the proposed decision making technique are shown in Figure 5.5.

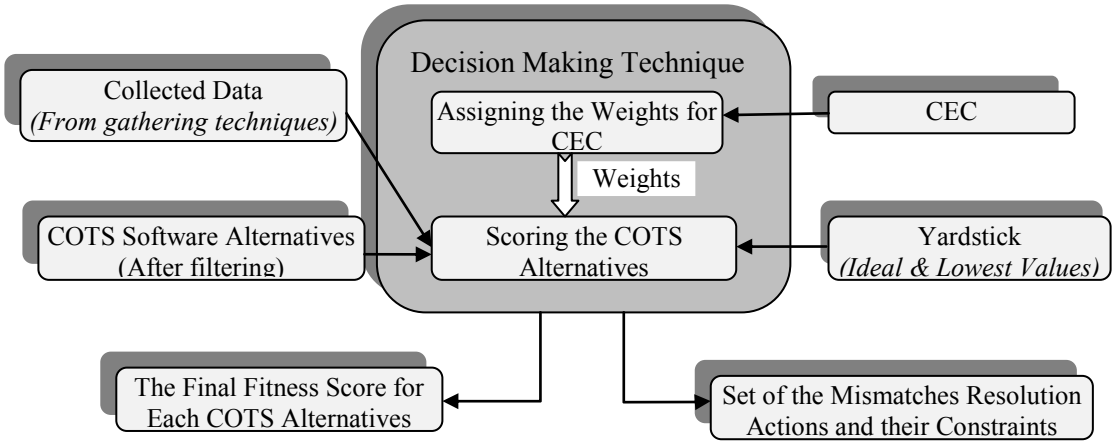


Figure 5.5. The Proposed Decision Making Technique

5.3.5.1 Assigning Weights for CEC

A weight should be assigned for each CEC criterion because it does not only give value for each criterion but also represents the relative importance or users' preferences. This is required because the evaluation criteria are not equal to its importance. For example, the reliability of the COTS software is often more important than maintainability because most vendors do support the COTS software maintainability. This makes it less important compared to the reliability of the COTS software. This task is carried out through four stages, which are: i) Constructing the

Pairwise Matrix, ii) Performing the Judgments of Pairwise Comparisons, iii) Synthesizing the Pairwise Comparison, and iv) Performing the consistency (Saaty, 1980).

1. Constructing the Pairwise Matrix

The pairwise comparisons are used to assign the weight for the criteria. The evaluation criteria are sorted in CEC in a full hierarchy structure where the criteria are distributed through several levels. Figure 5.6 shows an example from the CEC hierarchy structure.

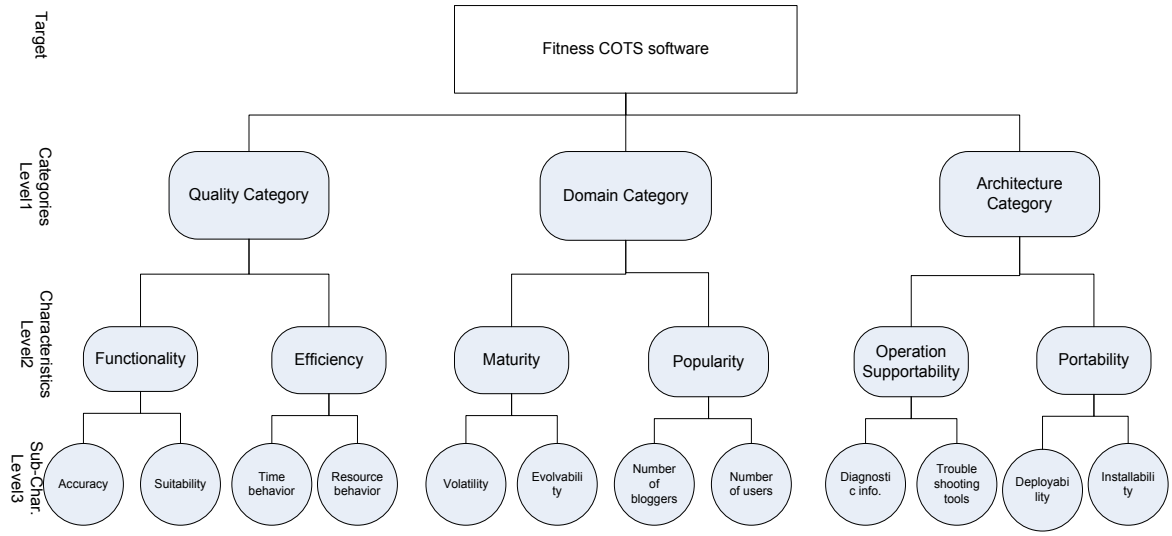


Figure 5.6. An Example of the CEC Hierarchy Structure

The sibling criteria at each level are compared in pairs to judge which of each criterion is preferred or important with respect to their parents. The pairwise comparison process sorts the sibling criteria of each level of the CEC in matrix of

two dimensions (square matrix) where the same criteria are sorted horizontally in the first row and vertically in the first column in the matrix, as shown in Figure 5.7. The criteria in the matrix is represented by ($C1 \ldots C_n$) and the relative importance degree of each C_i in the column compared to the C_j in the row is represented by (a_{ij}) with the constraints that $a_{ij} = 1/a_{ji}$ when $i \neq j$, and $a_{ii} = 1$ when $i = j$. See Figure 5.6.

Criteria	C_1	C_2	C_3	...	C_m
C_1	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	…	$a_{1,m}$
C_2	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	…	$a_{1,m}$
C_3	$a_{2,1}$	$a_{3,2}$	$a_{3,3}$	…	$a_{1,m}$
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
C_n	$a_{n,1}$	$a_{n,2}$	$a_{n,3}$	…	$a_{n,m}$

Figure 5.7. The Pairwise Matrix

At the CEC, the main five categories in the first level are compared with respect to the evaluation target using one pairwise comparison matrix, as shown in the Table 5.15.

Table 5.15

The Pairwise Comparison Matrix for Level One in CEC

Fittest category	Quality	Domain	Architectural	Operational environment	Vendor
Quality	1	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$
Domain	$1/a_{1,2}$	1	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$
Architectural	$1/a_{1,3}$	$1/a_{2,3}$	1	$a_{3,4}$	$a_{3,5}$
Operational environment	$1/a_{1,4}$	$1/a_{2,4}$	$1/a_{3,4}$	1	$a_{4,5}$
Vendor	$1/a_{1,5}$	$1/a_{2,5}$	$1/a_{3,5}$	$1/a_{4,5}$	1

In the second level, based on the five categories there are five pairwise comparison matrixes where the characteristics of each category are represented by one matrix

such as the quality category characteristics (functionality, reliability, efficiency, .. etc) are compared with each other with respect to the quality category using one pairwise matrix as shown in Table 5.16. The same procedure is applied for the other categories (domain, architectural, operational environment and vendor categories).

Table 5.16

The Pairwise Comparison Matrix of the Quality Category

Quality characteristics category	Functionality	Reliability	Efficiency	Usability	Testability	Maintainability
Functionality	1	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$
Reliability	$1/ a_{1,2}$	1	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$
Efficiency	$1/ a_{1,3}$	$1/ a_{2,3}$	1	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$
Usability	$1/ a_{1,4}$	$1/ a_{2,4}$	$1/ a_{3,4}$	1	$a_{4,5}$	$a_{4,6}$
Testability	$1/ a_{1,5}$	$1/ a_{2,5}$	$1/ a_{3,5}$	$1/ a_{4,5}$	1	$a_{5,6}$
Maintainability	$1/ a_{1,6}$	$1/ a_{2,6}$	$1/ a_{3,6}$	$1/ a_{4,5}$	$1/ a_{5,6}$	1

In the level three of the CEC, there are 21 matrixes to do the pairwise comparisons between the sub-characteristics. The pairwise comparison is done between the siblings with respect to their parents in each matrix. Table 5.17 summarizes the number of pairwise comparisons matrixes at each level of the CEC.

Table 5.17

The Pairwise Comparison Matrixes in CEC

CEC levels	**Number of pairwise matrixes**
Level 1	1
Level 2	5
Level 3	21

2. Performing the Judgments of Pairwise Comparisons

The pairwise comparison begins by comparing the relative importance of each two criteria in the matrix, e.g. *"is C_1 important than C_2 with respect to their parent? How much is it important ($a_{i,j}$)?"*. In order to determine the intensive importance of each pairwise comparison (a_{ij}), a fundamental scale of absolute numbers suggested by Saaty (1980) is used. This scale, as shown in Table 5.18, has been proven in practice and validated by decision problems experiments in helping the team developers to assign related importance to each pair of the ratio.

Table 5.18

Fundamental Scale for Pairwise Comparison (Saaty, 1980)

Intensive of importance	Definition
1	Equal importance
3	Moderate importance
5	Strong importance
7	Very strong importance
9	Extreme importance
2, 4, 6, 8	Intermediate judgment values

The number of the pairwise comparisons in each matrix is determined by the following formula:

Pairwise comparisons in each matrix = n (n-1)/2 …… (5.1)

where "n" is the number of criteria in the matrix.

For instance, by referring to the example in Table 5.16 which has six criteria (n=6), the identification of the pairwise comparisons number in the matrix is done by applying the previous formula: 6(6-1)/2 = 15 pairwise comparisons.

The judgments are decided based on the evaluation team. For more explanation, refer to the example in Table 5.15. In the first level of the CEC, the number of pairwise comparison is 5(5-1)/2 = 10. The pairwise comparisons should be conducted in this matrix with respect to the target of selecting the fittest COTS software. These are some of such comparisons:

- Is the "quality category" important than the "domain category"? Yes, it is a moderate importance than domain category (3), according to the scale in Table 5.18.

- Is "quality category" important than the "architectural category"? Yes, it is strongly important than the architectural category (5).

- Is the "architectural category" important than the "operational environment category"? No, the operational environment is moderately important than the architectural category (1/3).

The reciprocals are then assigned to other each of the pair-wise comparisons in the matrix, see Figure 5.8.

Selecting the fitness COTS	Quality	Domain	Architectural	Operational Environment	Vendor
Quality	1	a=3	5	3	5
Domain	1/3	1	3	1	3
Architectural	1/5	1/3	1	1/3	3
Operational Environment	1/3	1	3	1	3
Vendor	1/5	1/3	1/3	1/3	1

Figure 5.8. Example of Performing the Judgments Pairwise Comparisons

3. The Pairwise Comparison Synthesis

After the completion of assigning the evaluation team judgments in each matrix, the vector of weights is calculated. To do so, the Average Normalized Column method (ANC) is used (Hsiao, 2002). This method is conducted by dividing the elements in each column by the sum of that column in the matrix. The average of each resulting row is then calculated to obtain the weight for each criterion in the matrix. The vector of weight is calculated according to Ariff et al., (2012) by the following formula:

$$W_i = \frac{1}{n}\sum_{j=1}^{n}\frac{a_{ij}}{\sum_i^n a_{ij}}, i, j = 1, 2, \ldots n \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5.2)$$

where, *"a_{ij}"* is the value of the intensive of importance (from Table 5.18) of the pairwise comparison between C_i and C_j. *"n"* is the number of criteria in the matrix.

For instance, to calculate the weights for the quality in the previous example in Figure 5.8, formula 5.2 is used as follows:

$\sum_i^n a_{ij}$ = 1+1/3+1/5+1/3+1/5= 2.067.

$\frac{a_{ij}}{\sum_i^n a_{ij}}$ = 1/2.067=0.484

$\sum_{j=1}^{n}\frac{a_{ij}}{\sum_i^n a_{ij}}$= 0.484+0.529+0.405+0.529+0.333=2.281, finally, divide this sum by the number of criteria in the matrix (n=5) = 2.281/5 = 0.456, the result is the weight for the quality.

4. Performing the Inconsistency Test

Since the pairwise comparison process is carried out through subjective judgment, the inconsistency could be raised. Thus, the consistency verification is carried out to ensure that the judgments by the evaluation team are consistent. The first step in measuring the degree of consistency in each matrix is by determining the consistency ratio (CR). CR is calculated to measure whether the consistent judgments are relative to the large samples of purely random judgments. If the value of CR exceeds 0.10, the judgments are considered unreliable (Saaty, 1980). In this case, review is required in order to accept the judgments. CR is the ratio of consistency index (CI) to random index (RI) for the same order matrices. CR is calculated using the following formula:

$$CR = CI/RI \quad \ldots\ldots\ldots\ldots (5.3)$$

where "CI" is calculated using the following formula:

$$CI = (\lambda_{max} - n)/ (n - 1) \quad \ldots\ldots\ldots\ldots (5.4)$$

where *"n"* is number of criteria in the matrix, and λ_{max} is the maximum eigen value of the matrix.

λ_{max} can be calculated by following steps:

1) Multiply the summation of each column in the matrix by the weight vector and obtaining the new vector.

2) Divide all the elements of the weighted sum matrices or new vector by their respective weight vector element.

3) Find out the average of these values to obtain λ_{max}.

Choosing the suitable value of RI is based on the size of the matrix (n) as stated in Table 5.19. The RI value is developed by Saaty and his colleagues at the Oak Ridge National Laboratory by generating random matrices and calculating the mean of CI. Once identified the RI value, the CR can be calculated. According to Saaty, the judgment matrix is considered as inconsistent when the CR>0.1. This means that the judgments in that matrix need to be reviewed and improved.

Table 5.19

Random Index

Size of Matrix (n)	1	2	3	4	5	6	7	8	9	10	11	12
Random Index (RI)	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.58

It is worthwhile to mention that the matrixes judgments are carried out by the evaluation team that include experts, users, and technicians during their meeting using the JAD technique in making the most appropriate judgments.

5.3.5.2 The COTS Alternatives Scoring

The COTS alternative scoring is the process of estimating each COTS alternative against the evaluation criteria in order to measure the degree of matching between COTS alternatives and each criterion in the CEC. This process is carried out by the evaluation team. In the traditional way, the scoring is a subjective estimation because it is based on the level of evaluators' experience, which may create prejudice. Thus, the subjective estimation often does not assign accurate score that reflect on the

overall fitness of the COTS software which exist in the AHP technique (Mohamed et al., 2008).

The COTS mismatches problem is addressed and the solution is proposed during this task. Handling the mismatch problem in the evaluation and selection process is done by detecting these mismatches, identifying the solutions, and considering their constraints (i.e. effort and cost). These mismatches are resolved by filing the gaps between the COTS software alternatives and user requirements. In this task, the proposed methodology for handling COTS mismatches and calculating the final score is presented.

The COTS mismatches handling is a process of identifying and analyzing mismatches, and providing required information for calculating the final score of the COTS software against the evaluation criteria. This will eventually lead to the making of appropriate decision for selecting the COTS software. Handling the COTS mismatches consists of five stages as shown in Figure 5.9.

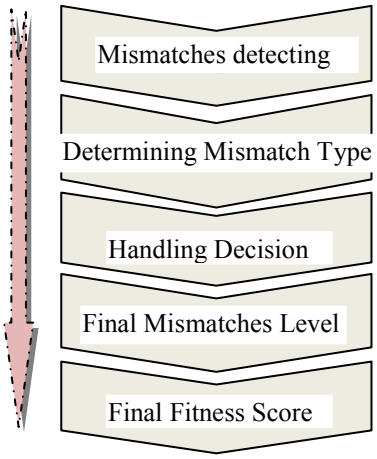


Figure 5.9. Stages of Handling the COTS Mismatches

1. Detecting the COTS Mismatches

The COTS mismatches are detected when the COTS software features are sorted against the evaluation criteria in the low level of the CEC. The COTS software alternatives (A) and the criteria (C) are evaluated using two dimensional matrixes, as shown in Figure 5.10.

criteria / COTS alternatives	C_1	C_2	C_3	…	C_n
A_1	√	√	×	…	√
A_2	×	√	√	…	×
A_3	×	×	×	…	√
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
A_m	√	√	√	…	×
√ : COTS alternative (A) match the criterion (C) × : COTS alternative (A) mismatch the criterion (C)					

Figure 5.10. Mismatches Detection Matrix

The COTS mismatches are identified by comparing the COTS software values against the ideal and lowest values in the yardstick. A mismatch occurs when the COTS software value does not achieve the ideal value of the criteria in the yardstick.

2. Determining Type of the COTS Mismatches

As previously mentioned, the mismatches between the COTS software and user requirement are classified into four types: fail, partially, fulfill, and extend (hurtful, helpful, and neutral extend match) (Alves, et al., 2005, Mohmad, 2008). For more details refer to section 2.3.4. In general, the COTS mismatches are determined at each criterion by comparing the value of the COTS software to the yardstick

thresholds values. The type of COTS mismatches is identified according to the following scenarios:

- Fail-match (or full mismatch) is raised when the value of COTS software at the certain criterion is less than the lowest value in the yardstick.
- Partially-match (or partially mismatch) occurs when the value of the COTS software is less than the ideal value and equal or more than the lowest value of the yardstick at certain criterion.
- Fulfill-match (or zero-mismatch) occurs when the value of COTS software is equal to the ideal value in the yardstick at certain criterion.
- Over-match (positive-mismatch) occurs when the value of the COTS software is more than the ideal value in the yardstick at certain criterion.

These scenarios of the types of COTS mismatches are summarized in Table 5.20.

Table 5.20

Scenarios of Identifying the Types of the COTS Mismatches

Scenarios	**Types**
$X_{c1} < L_{c1}$	Fail-match (or Full-mismatch)
$L_{c1} \leq X_{c1} < I_{c1}$	Partially-mismatch
$X_{c1} = I_{c1}$	Fulfil-match (or zero-mismatch)
$X_{c1} > I_{c1}$	Over-match (or positive-mismatch)
X: the COTS value at criterion c_1 L: the lowest value defined in the yardstick at criterion c_1 I: the ideal value defined in the yardstick at criterion c_1	

The decision of handling the COTS mismatch depends on the type of the mismatches. Among the decisions of dealing with the COTS mismatch are resolve and tolerate. The following section elaborates on these decisions.

3. The COTS Mismatches Handling Decisions

The decision on how to handle any mismatch should be made during the evaluation process and before selecting the COTS software. In this research, three decisions are considered for handling the mismatches:

i) Resolving the COTS mismatch.

ii) Tolerating the COTS mismatch.

iii) Rejecting the COTS alternative.

Figure 5.11 shows the mechanism of choosing the mismatch handling decision. The mechanism for selecting the right decision depends on the types and levels of the COTS mismatches. Therefore, the evaluation team should estimate whether the identified mismatch is significantly effective or not on the fitness of the COTS software. In addition, the mismatch needs to be examined to determine whether it can be resolved or rejected. Otherwise, the evaluation team will choose to tolerate the mismatch if there is no significant effect on the fitness of the COTS software.

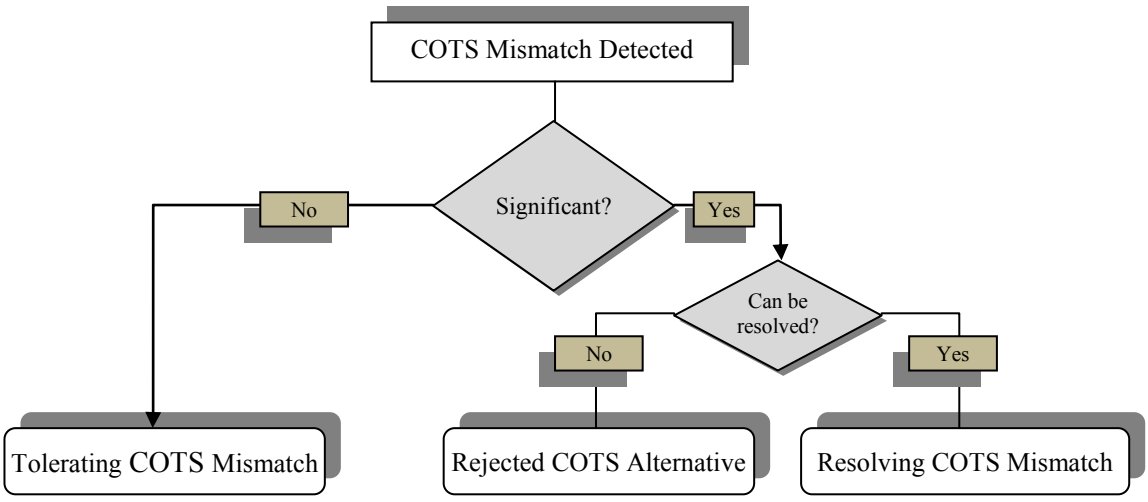


Figure 5.11. The Decision of Handling COTS Mismatch

The types of the COTS mismatches and the potential mismatch handling decisions are summarized in Table 5.21.

Table 5.21

Potential Decisions for Each Type of COTS Mismatches

Types		**Possible Decision**
Fail-match (full-mismatch)		Resolving , COTS rejected
Partially-match		Resolving, tolerated, COTS rejected
Fulfil-match (zero-mismatch)		No handling decision
	Helpful	Resolving (adding it as a part of criteria)
Over-match (positive-mismatch)	Hurtful	Resolving. Tolerated, COTS rejected
	Neutral	Tolerated

The main decisions for handling the COTS mismatches are further explained in the following sections.

A. Resolving COTS Mismatches

The decision of resolving the COTS mismatch is chosen when the COTS mismatch has importance effect on the COTS fitness and when there is possible solution. However, resolving the COTS mismatches has three directions: i) yardstick adjustment, ii) or customizing COTS software, iii) or some time using both previous directions (adjusting the yardstick and COTS customization). Selecting one of these directions is based on its costs, time, efforts and risks on the organization. Figure 5.12 shows the three directions for solving the COTS mismatches.

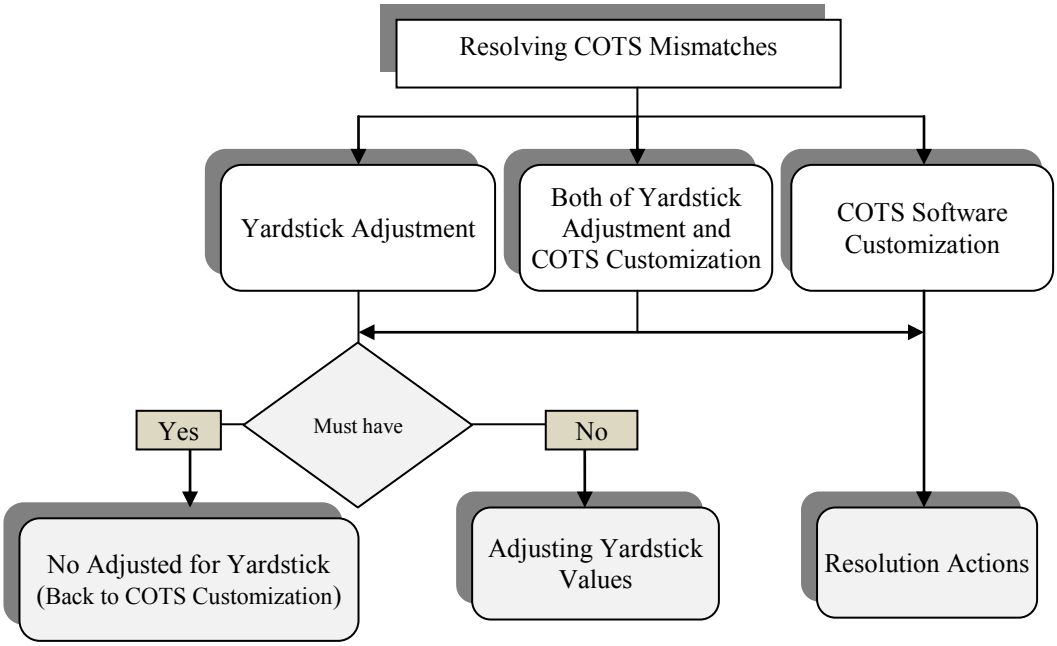


Figure 5.12. Resolving the COTS Mismatches Directions

i. Yardstick Adjustment

The yardstick adjustment is represented by modifying the yardstick thresholds values to meet the COTS software features. It is conducted through the evaluation process when the value of the attribute that causes the mismatch can be reviewed and updated to fill the gap of the COTS software. The attributes in the yardstick are classified into two types: i) “must have” and ii) “nice to have”. The first type, “must have”, cannot be changed because this kind of attributes mostly represents the key requirements of the user. If they are changed, the scope or objectives of project might also change. The other type, “nice to have”, is more flexible and easy to change without affecting the key requirements of the user. However, it is difficult to find the COTS software that exactly achieves all the yardstick thresholds values. Therefore, these values should be more flexible to meet the COTS software features in order to mitigate the mismatches between them.

ii. The COTS Software Customization

Typically, the evaluation team does not know yet which COTS software is going to be selected and therefore customizing all COTS software alternatives before selecting the best one requires a lot of efforts, costs, and time. Thus, resolving the mismatches by modifying the selected COTS software alternative is only possible after completing the evaluation and selection process, specifically in the adaptation and integration phases in the CBD. In this research, the COTS customization is performed during the evaluation and selection process by identifying the appropriate resolution action (i.e. scripting, add-on, and others). The customization is to fill the mismatch (gap) with the yardstick values, and identify the required resource constraints (i.e. cost, effort, and time) in order to compute the final score for each COTS alternative.

The COTS software is customized by performing a set of resolution actions such as add-on, scripting, and API. For more details refer to section 2.3.4. In this context, there are several constraints associated with these resolution actions that should be considered when calculating the final score for each COTS software alternative, which are: cost, effort, time, and risk. Each of these constraints has maximum value that should not exceed the project constraints (budget and time). Therefore, any resolution action to customize the COTS software should be applied with less cost, effort, time, and risk. Table 5.22 presents the sorted COTS mismatches as well as their resolution actions and constraints.

Table 5.22

The COTS Mismatches Information

Alternatives	Mismatches	Resolution action	Cost	Effort	Time	Risk
COTS 1	M_1	A_1	$Cost_1$	$Effort_1$	$Time_1$	$Risk_1$
	M_2	A_2	$Cost_2$	$Effort_2$	$Time_2$	$Risk_2$
	.	.	.	.	.	.
	.	.	.	.	.	.
	.	.	.	.	.	.
	M_n	A_n	$Cost_n$	$Effort_n$	$Time_n$	$Risk_n$
COTS 2	M_1	A_1	$Cost_1$	$Effort_1$	$Time_1$	$Risk_1$
	M_2	A_2	$Cost_2$	$Effort_2$	$Time_2$	$Risk_2$
	.	.	.	.	.	.
	.	.	.	.	.	.
	.	.	.	.	.	.
	M_n	A_n	$Cost_n$	$Effort_n$	$Time_n$	$Risk_n$

As shown in the previous Table, each COTS software may has many mismatches (M_n). Each of these mismatches can be solved using specific resolution action (A_n). The resolution action is associated with several constraints such as cost, effort, time, and risk. This information is used to identify the accurate score of each COTS alternative.

iii. The Yardstick and COTS Adjustment

In some cases, customization of the COTS software is not enough to solve the mismatches problem. Therefore, the yardstick values adjustment is also required to get the best solution. The values in the yardstick can be updated to allow few modifications in the COTS software to ensure that the resolution action is easier and more efficient in terms of cost, effort, time, and risk. This kind of resolution is used to mitigate the costs of modifying the COTS software or the risks of changing some of the yardstick values.

B. The COTS Mismatch Tolerance

The other option of the COTS mismatches handling decision is tolerance. Typically, this type of decision is selected when the risk of tolerating the COTS mismatch is less than the cost of solving it. In other word, if the COTS mismatch has no significant impact on the fitness of the COTS alternative, it can be ignored.

C. The COTS Alternative Rejection

If the mismatch has an important impact on the software fitness and cannot be resolved by customizing the COTS software or adjusting the yardstick, the COTS alternative that causes this mismatch will be rejected. Rejection decision refers to the exclusion of the COTS alternative that causes the unsolved mismatches from the evaluation process.

Table 5.23 summarizes the possible decisions of handling the COTS mismatches.

Table 5.23

Conditions of the COTS Mismatches Handling Decisions

	Handling Decisions	Conditions
Resolving decision	Adjusting the yardstick	- Significantly impact of COTS mismatch. - Peripheral (nice to have) values - COTS customizing requires high cost, effort, time, or risk.
	COTS customization	- Significantly impact of COTS mismatch. - Cannot adjust the yardstick (must have values). - Can be customized by existing resolution actions (ex. Add-on, API). - The resource constrains (ex. cost) with the project constrains (ex. budget).
	Combined of adjusting yardstick and customizing COTS software	- Adjusting the yardstick does not enough to fill the gap with COTS mismatches. - Customizing COTS software does not enough to solve the mismatches. - Possibility of few change in the yardstick mitigate

	the costs or risks of COTS customization and vice versa.
Tolerated COTS mismatch	- No significantly impact of COTS mismatch. - Resolving COTS mismatch required high cost and time.
COTS Rejection decision	- Significantly impact of COTS mismatch. - Cannot resolve COTS mismatch - Cannot adjust the yardstick values (must have values). - Resolving COTS mismatch required high cost, effort, time, or risk.

4. The Final Mismatching Level (FMML)

The matching level (ML) value is a quantitative estimation that represents the degree of matching between the COTS features and each criterion in the yardstick. The mismatches values can easily be identified based on the ML value. ML is defined as a matrix that uses the ratio scale normalized to the range from 0 to 1.

Precisely, let $A_k = \{a_1, a_2, a_3, \ldots, a_z\}$ be a set of COTS software alternatives. Each of the COTS software alternative (A_k) has a set of features $F_{ak} = \{f_1, f_2, f_3, \ldots, f_n\}$. These features are compared against the evaluation criteria $C_s = \{c_1, c_2, c_3, \ldots, c_m\}$.

ML value is defined as the following metric:

$$ML_{(fi, cj)} = \begin{cases} 1 & \text{iff } f_i \text{ (full match) } c_j \\ 0 - 1 & \text{iff } f_i \text{ (partially match) } c_j \\ 0 & \text{iff } f_i \text{ (fail to match) } c_j \end{cases} \quad \ldots\ldots\ldots\ldots (5.5)$$

where ML is equal to 1 when a feature (f_i) fully matched the criterion (c_j) (zero-mismatch). The value of ML will be between 0 and 1 when there is a partially matching between the feature (f_i) and criterion (c_j) (partially-mismatch), and fail or no matching if f_i and c_j is represented by 0 (full-mismatch). There are different kinds of metrics that can be used to measure the attributes in the CEC such as presence,

level, integer, and percentage (ratio). The rule for mapping these metrics to the ML range from 0 – 1 is required. The matches of the attributes are represented by the following linear function (Alves et al., 2005):

$ML_{cij}(X) = a\,X + b$ …… (5.6)

where "c_{ij}" represents the criterion; "X" represents the COTS value; "a" and "b" represent the constants.

The previous formula can be used to map or normalize different kinds of metrics of the ML ranging from 0 to 1:

$$ML_{(Xi,\,cj)} = \begin{cases} 1 & X >= \text{ideal value in the yardstick} \\ a\,X + b & \text{lowest value} =< X < \text{ideal value} \\ 0 & X < \text{lowest value} \end{cases} \quad \text{……(5.7)}$$

where "*a"* and "*b"* represent the constants, which can be calculated as the following:

$a = 1/(I_{c\,i} - L_{c\,i})$ ……..……..……(5.8)

$b = -\,L_{c\,i} / (I_{c\,i} - L_{c\,i})$ ……..… (5.9)

where "I" is the ideal value of criterion (c_i), and "L" is the lowest value of criterion (c_i) defined in the yardstick.

For Example, the following table includes several criteria that represent different metrics from the low level of CEC, the yardstick (ideal & lowest) values, and the collected COTS values at these criteria.

Table 5.24

An Example to Calculate ML

Criteria	Ideal Value	Lowest Value	COTS value (X)	Metric	ML
Coverage	95%	70%	90%	Ratio (%)	0.8
Excess	95%	85%	95%	Ratio (%)	1
Persistence	1	0	0	Presence (1/0)	0
Error handling	1	1	1	Presence (1/0)	1
Disk utilization	1 GB	3 GB	3 GB	Integer value	0
Response time	1(excellent)	3 (good)	2 (very good)	Level	0.5

According to formula 5.7, the $ML_{(Coverage)}$ was calculated as following:

The COTS value (X) at coverage criterion is (90%) which is less than ideal value (95%). This means that there is a partially match (according to formula 5.5) and the ML can be calculated using formulas 5.6, 5.8, and 5.9:

$ML_{(x=90\%,\ coverage)} = a(90\%)+b$ …….. (Formula 5.6)

$a= 1/ (95 - 70) = 0.04$ …… (Formula 5.8)

$b= - (70) / (95 - 70) = -2.8$ …… (Formula 5.9)

$ML = (0.04) (90) – 2.8 = 0.8$

$ML_{(x=90\%,\ coverage)} = 0.8$ (partially match)

The rest ML values of other criteria were calculated using the same procedure.

Next is the calculation of the mismatch level (MML) between the COTS feature (f_i) and the criterion (c_j) by using the following formula:

$MML_{(fi,\ cj)} = 1 – ML$ …… (5.10)

Identifying the MML is not enough to provide accurate selection decision in choosing the appropriate COTS software. There are still many issues need to be considered when making a selection decision. Firstly, identifying appropriate solution or resolution action for each significant COTS mismatch based on the

decision of handling the mismatch. There are many kinds of resolution action that can be used to fill the mismatches between the COTS features and user requirements such as developing the custom code, API, negotiating with user to change certain requirements, and others. The suitable resolution actions are determined by the evaluation team meeting using the JAD technique.

Each resolution action has a set of constraints that appear during its development, implementation, and testing. These constraints are: i) costs of the resolution, ii) the efforts to develop, apply, and test the resolution, iii) the time of developing and testing the resolution, and v) the risks of applying the resolution on the COTS software and final system. These constraints play important role in supporting accurate decision making of selecting the most appropriate COTS alternative that can be customized and integrated within the current limited resources (e.g. budget, and time) of the CBD project. For instance, the COTS software that has many mismatches that can be solved efficiently in terms of cost, time, effort, and risk is more fit than the COTS software that has few mismatches that can be solved with high risks and costs.

The Likert scale of five points is used to estimate the above constraints, as shown in Table 5.25. The value of each resolution constraint is used to calculate the final mismatch level of the COTS software at each criterion.

Table 5.25

Representation of the Resolution Action Constraints Values

Value	Meaning
1	Very Low
2	Low
3	Average
4	High
5	Very High

The final mismatch level (FMML) for the COTS alternative at each criterion is calculated by using the following formula:

$$FMML_{(fi, cj)} = (MML_{(fi, cj)} * (c + t + e + r) / 4) / 5 \quad \text{.....(5.11)}$$

where,

FMML: the final mismatch level for the COTS feature (f_i) and the criterion (c_j)
MML: the mismatch level between the COTS feature (f_i) and the criterion (c_j)
"c": the costs; "t": the required time; "e": the required efforts; and "r": the level of potential risk of resolution action for solving the mismatch between the COTS feature (f_i) and the criterion (c_j).

The previous formula aims to calculate the final mismatch level between the COTS feature (f_i) and the criterion (c_j) by considering the constraints (*c, t, e,* and *r*) of the resolution action. The constraints are important for providing accurate score that reflect the real value of the COTS mismatch level.

For Example, by referring to Table 5.24 that includes set of criteria and their ML values, the MML and FMML values are calculated as following:

Firstly, calculating MML for each criterion using formula 5.10 as following:

$MML_{(coverage)} = 1 - 0.8 = 0.2$

$MML_{(excess)} = 1 - 1 = 0$

$MML_{(persistence)} = 1 - 0 = 1$

$MML_{(error\ handling)} = 1 - 1 = 0$

$MML_{(disk\ utilization)} = 1 - 0 = 1$

$MML_{(response\ time)} = 1 - 0.5 = 0.5$

Secondly, calculating FMML for each criterion using formula 5.11 as following:

$FMML_{(coverage)} = ((c=3 + t=2 + e=3 + r= 1) / 4) / 5 = (9 / 4) / 5 = 0.45$

(The values of c, t, e, and r is selected from table 5.25 for each resolution action)

$FMML_{(excess)} = ((1 + 2 + 2 + 3) / 4) / 5 = 0.40$

$FMML_{(persistence)} = ((4 + 1 + 3 + 3) / 4) / 5 = 0.55$

$FMML_{(error\ handling)} = ((3 + 3 + 2 + 1) / 4) / 5 = 0.45$

$FMML_{(disk\ utilization)} = ((2 + 1 + 1 + 1) / 4) / 5 = 0.25$

$FMML_{(response\ time)} = ((5 + 4 + 3 + 5) / 4) / 5 = 0.85$

5. The Final Fitness (Matching) Scoring (FFS)

After determining the weights of all criteria in the CEC (section 5.3.5.1) and identifying the final mismatch level (FMML) for each COTS software alternative, the next step is to consider the total fitness or matching estimation (scored) process of the COTS software alternative against the CEC. The process goes through the following steps:

1) Calculating the final matching level (FML) of the COTS software at each criterion in level 4 using the following formula:

$FML_{(fi, cj)} = 1 - FMML_{(fi, cj)}$ ……..……(5.12)

2) Calculating the FML for each criterion in level 3 of the CEC by computing the average of FML values for its' child attributes in level 4 using the following formula:

$$FML_{C.L3} = \frac{\sum_1^m FMLi}{m} \quad \text{..... (5.13)}$$

where $FML_{C.L3}$ is the final matching level for each criterion in level 3 of the CEC; "*m*" is the number of child of the level 3 criterion; and "FML_i" is the final matching level for each child.

3) Calculating the total fitness score for each COTS software alternative against the CEC (starting from level 3 to level 1). This calculation is carried out using the following formula:

$$\text{Parent-Score} = \frac{\sum_{i=1}^n Wi * FMLi}{\sum_{i=1}^n Wi} \quad \text{..... (5. 14)}$$

where, "*n"* is the number of siblings that share the same parent; "W_i" is the weight of criterion (c_i); and "FML_i" is the final matching level at (c_i).

The previous formula is used to calculate the parent-score (FML) in the CEC starting from the low level (level 3) and aggregating the weighted scores upwards until reaching the root (the main target that represent the final fitness score of the COTS software overall criteria is called FFS).

For Example, simple part of CEC is used to clarify how to calculate the FFS, see Figure 5.13, as following:

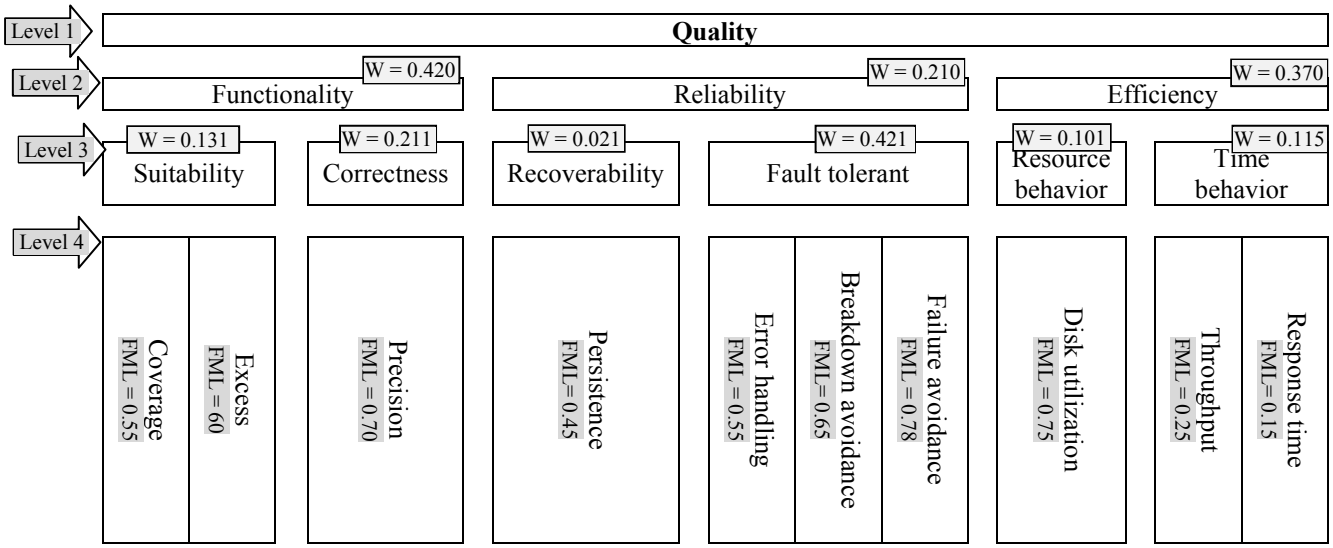


Figure 5.13. The Example to Calculate FFS

Firstly, calculating FML according to the formula 5.12 as following:

$FML_{(coverage)} = 1- (FMML_{(coverage)}= 0.45) = 0.55$

$FML_{(excess)} = 1 – (FMML_{(excess)}= 0.40) = 0.60$

The procedure was used to calculate the FML values for all rest criteria in the level 4 in the example.

Secondly, to calculate the FML values for criteria in level 3 the formula 5.13 was used where each criterion in level 3 used its' child in level 4 as following:

$FML_{(suitability)} = \Sigma (FML_{(coverage)} + FML_{(excess)}) / 2 = (0.55 + 0.60) / 2= 0.575$

$FML_{(correctness)} = FML_{(precision)} / 1 = 0.70 / 1 = 0.70$

$FML_{(recoverability)} = FML_{(persistence)} / 1 = 0.45 / 1 = 0.45$

$FML_{(fault\ tolerant)} =\Sigma (FML_{(error\ handling)} + FML_{(breakdown)} + FML_{(failure\ avoidance)}) / 3 = (0.55 + 0.65 + 0.78) / 3 = 0.66$

$FML_{(resource\ behavior)} = FML_{(disk\ utilization)} / 1 = 0.75$

$FML_{(time\ behavior)} = \Sigma (FML_{(response\ time)} + FML_{(throughput)}) / 2 = (0.15 + 0.25) / 2 = 0.20$

Thirdly, to measure the FML for the criteria in the level 2, the formula 5.14 is used. To apply this formula, the weights of criteria that calculated in first step of the

decision making technique are required with FML values of the child of each criterion in level 3. In this example, let assume the weights values as shown in figure 5.13. The formula 5.14 is applied as following:

$$\text{Parent-Score}_{(\text{functionality})} = (\Sigma(w_{(\text{suit.})}=0.131 * \text{FML}_{(\text{suit.})}=0.575 + w_{(\text{correct.})}=0.211 * \text{FML}_{(\text{corr.})}= 0.70)) / (w_{(\text{suit.})} = 0.131 + w_{(\text{corr.})} = 0.211) = .652$$

$$\text{Parent-Score}_{(\text{reliability})} = (\Sigma(w_{(\text{recov.})}=0.021 * \text{FML}_{(\text{recov.})}=0.450 + w_{(\text{fault.})}=0.421 * \text{FML}_{(\text{fault.})}= 0.660)) / (w_{(\text{recov.})} = 0.021 + w_{(\text{fault.})} = 0.421) = 0.649$$

$$\text{Parent-Score}_{(\text{efficiency})} = (\Sigma(w_{(\text{res.})}=0.101 * \text{FML}_{(\text{res.})}=0.750 + w_{(\text{tim.})}=0.115 * \text{FML}_{(\text{tim.})}= 0.20)) / (w_{(\text{res.})} = 0.101 + w_{(\text{tim.})} = 0.115) = 0.458$$

Finally, applying the same procedure to calculate the score that represents the value of FFS at quality root in the example as following:

$$\text{Parent-Score}_{(\text{quality})} = (\Sigma(w_{(\text{fun.})}=0.420 * \text{FML}_{(\text{fun.})}=0.652 + w_{(\text{rel.})}=0.210 * \text{FML}_{(\text{rel.})}= 0.649) + w_{(\text{eff.})}= 0.370 * \text{FML}_{(\text{eff.})}= 0.458) / (w_{(\text{fun.})} = 0.420 + w_{(\text{rel.})} = 0.210 + w_{(\text{eff.})} = 0.370) = 0.58 \text{ (FFS)}.$$

The third step in the proposed decision making is repeated for all COTS alternatives. After its determination, the FFS is sorted from the high to low values accordingly. The main advantages of the proposed decision making include:

- Selecting the most appropriate COTS software that has a high FFS.

- Providing a sorted list of the COTS alternatives based on their fitness against the evaluation criteria. The alternative with the highest score will be the fittest software that fulfills user requirements. This score is located on the top of the list, while the second high alternative will be listed as the second highest score in the list and so on.

- Providing feedback to vendors about the weakness of their products based on the related information such as the mismatches values, solutions and constraints.

- Providing set of information about the selected COTS software related to its mismatches, solutions, costs and risks. In addition, the information related to the vendors is also provided. For example, whether the vendors are providing software support and upgrade to the developers in the next phases of the CBD such as the integration phase.

- The previous information can be stored and retrieved in order to learn from it for similar future projects.

5.3.6 Evaluation Processes

The evaluation process consists of a set of related activities and tasks that are performed in order to achieve the evaluation objectives. All of the previous components are performed, integrated, and consolidated in three main processes as shown in Figure 5.14:

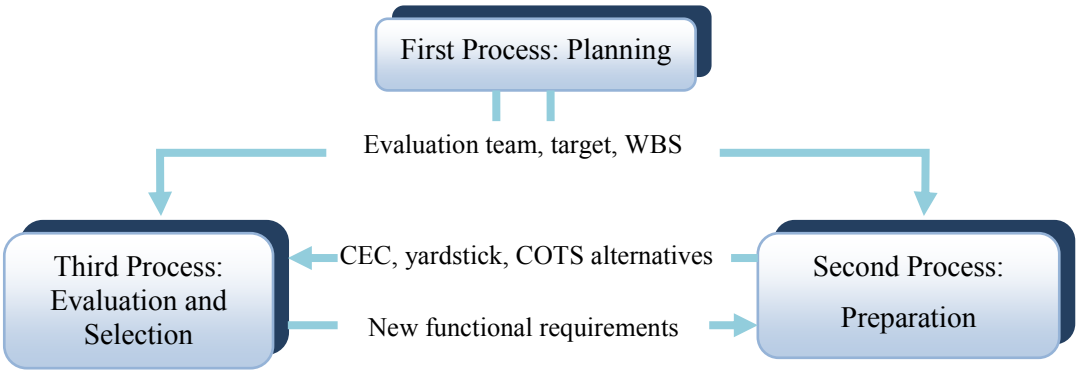


Figure 5.14. The Evaluation Processes

1. **Planning process**: This is the first activity in the evaluation and selection process, which includes: the evaluation target definition, the evaluation team formation, and work breakdown structure creation.
2. **Preparation process**: This is also called the initializing or establishing process. In this process, the functional requirements are gathered, the COTS software alternatives are identified and the yardstick values are defined.
3. **Evaluation and selection process**: This is the core activity of the COTS software evaluation and selection process. In this process, the COTS alternatives are estimated against the evaluation criteria and the mismatches between them are handled in order to make an appropriate decision for selecting the fitness of COTS alternative.

5.3.6.1 Planning Process

The COTS-ESF is started by the planning process. It is an important process because the effort spent in the planning can save countless hours of confusion and rework in the subsequent processes. However, different types of COTS software (simple or complex) and different user expectations on new software make the planning process

for each COTS software evaluation differs. The planning process shares common activities to be completed. These activities are shown in Figure 5.15.

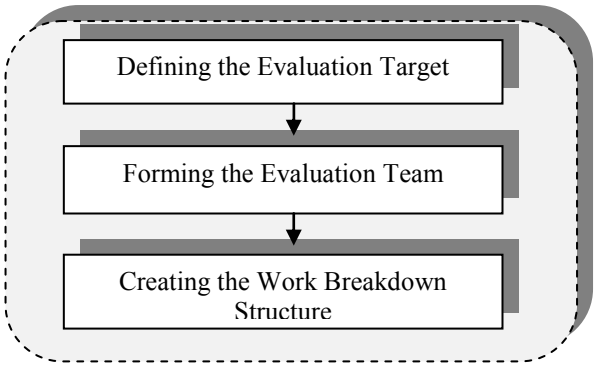


Figure 5.15. Planning Activities

A. Defining the Evaluation Target

The software components used in this research are defined in section 5.3.1. Therefore, the target software should be embraced under the COTS concept to be successfully evaluated by this framework. In addition, defining the target means that the target COTS software and its scope should be defined, described, and delimited explicitly. Furthermore, it is necessary to study and delimit the main objectives under the evaluation. Defining the evaluation target plays an important role to identify the potential evaluators who can deal with the target COTS software, as well as to help estimating the project constraints (e.g. budget and time).

B. Forming the Evaluation Team

The activity of gathering the right people (stakeholders) and getting them to work together for the benefit of the project is called forming the evaluation team. Many

elements should be considered when selecting the team members. The main elements include the team size, experiences (e.g. technical expert and domain expert), skills (e.g. programming and analysis skills), cooperation, and commitment to the project goals. In this context, the evaluation team should have the following kind of members:

i. The management or leader to direct and manage the team to achieve the goals.

ii. Expert in the domain of the software under evaluation.

iii. Several technical people such as the software developer, software analyst, and software architect.

iv. The end-user of the targeted COTS software.

The process or mechanism of recruiting good members in the team is by identifying the relevant people (stakeholders) inside and outside the organization. The recruitment is also based on the targeted project besides the previous elements. The identified candidates will then be contacted to check on their interest in the project. The preliminary information is vital in selecting the right person to join in the team.

Ultimately, after selecting the team members, the goals of the team should be defined and the tasks should be assigned by interacting with the members. In addition, the methodology of how the team members interact and work together should also be defined as well as the required resources such as time, budget, and

computer facility. When formation is completed, the team starts identifying and listing the potential stakeholders for the evaluation project, and establishing the Work Breakdown Structure.

C. Creating the Work Breakdown Structure (WBS)

WBS has an important role for the success of a project. It provides an overview and mission of the project. Once the WBS is completed and circulated, it becomes easier to implement and understand the whole picture of the project. The WBS includes most of the preliminary project elements such as the scope statements, which describe the main issues related to the project. It also provides assistance in managing changes to the project's scope during the execution. As an inclusive overview of the project, the WBS allows all parties (manager, evaluation team, stakeholders) to reach an agreement and document the major aspects of the project such as the goals, constraints, scope, and deliverables. It supports the decision making and it is often used as a communication tool.

The WBS is created by the evaluation team to define the scope and constraints of the evaluation. There are many issues included in the WBS depending on the project. Among the common issues shared by most of the projects are goals, scope, names of evaluation team and their responsibilities.

5.3.6.2 Preparation Process

The preparation process is also known as the pre-evaluation process, which refers to the preparing and providing required information for carrying out further evaluation

in the subsequent process. The primary objective of this process is to identify the functional requirements that are driven from the user requirements and organization constraints. It also aims to identify the potential COTS software alternatives that achieve the main target of the project as well as to define the yardstick values (ideal and lowest values).

The main inputs for this process are stakeholders, functional requirements sources and COTS sources, while the outcomes are an initial list of the COTS software alternatives, and yardstick. Typically, this process consists of three activities: defining functional requirements, searching the potential COTS alternatives, and defining the yardstick. These activities and their relationships are illustrated in Figure 5.16.

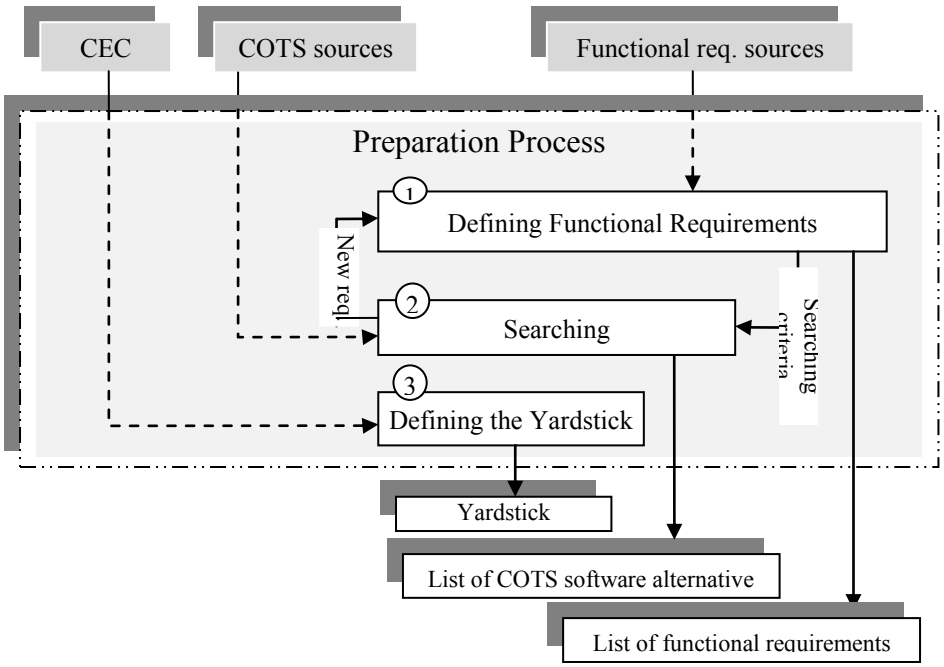


Figure 5.16. The Activities of the Preparation Process

A. Defining the Functional Requirements

This activity aims to identify the functional requirements that will support the identification of the COTS software alternatives. These requirements can be derived from several resources such as the system constraints, user/stakeholders expectations, COTS software features, previous evaluation cases, and operational environment. Findings of the empirical study show that the documents review and brainstorming session are the most common techniques used to identify the user's requirements. Therefore, these techniques are applied through the use of the JAD technique (refer to section 5.3.4).

B. The COTS Software Searching

This activity aims to identify the potential COTS software alternatives that match the identified functional requirements. The search is derived from the search criteria that have been defined based on the functional requirements. Typically, these criteria are included as the required main functionality of the COTS software searching as well as some of the key constraints (e.g. COTS software should be compatible with specific operating system such as WINDOWS). Generally, these criteria should be defined broadly to ensure that the search not only covers the depth but also the breadth in order to find all of the COTS potential candidates.

The COTS searching consists of three steps, as shown in Figure 5.17, which include:

1. Identifying the COTS software alternatives: It aim is to identify the COTS software that meets the search criteria (functional requirements). Identifying

the COTS software alternatives can be conducted through the COTS inventories search, market surveys, and internet search.

2. Identifying the COTS information: The preliminary relevant COTS information that should be identified during this step should include the following information: name of the alternative, primary characteristics (e.g. price, main functions, and basic vendor information), and reference of the COTS alternative source (e.g. the company name, address, website, etc).
3. Preparing the initial list of the COTS alternatives: This activity will generate the list of all potential COTS software with their relevant information.

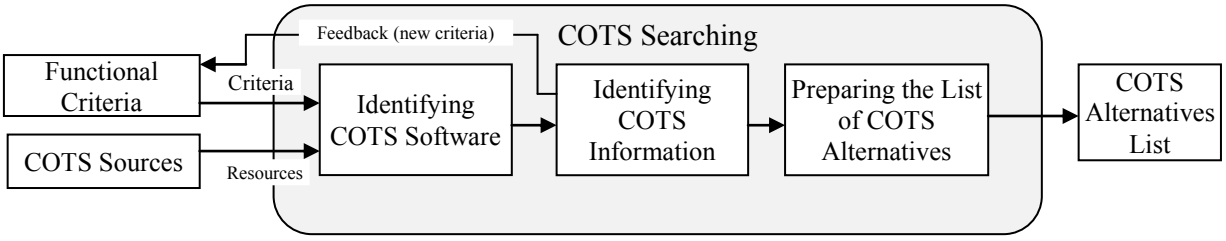


Figure 5.17. The COTS Searching

It is important to note that some extra functionality offered by some of the COTS alternatives and might be accepted by the evaluation team will be added to the functional criteria as a feedback. Finally, the main output of this activity is the general list of the potential COTS software with their relative information.

C. Defining the Yardstick

Defining the yardstick is the activity of assigning the ideal and lowest value for each attributes in the CEC (refer to section 5.3.3). The ideal values defined by this activity

play a vital role in identifying, classifying, and calculating the COTS mismatches levels in order to calculate the final score for each COTS alternative. In addition, the lowest values defined through this activity are used to filter out the COTS alternatives that fail to achieve these values. The filtering is required to decrease the numbers of COTS alternatives before applying the synthesis technique. Defining the yardstick activity is carried out by the evaluation team members that applied the JAD technique in assigning the accurate ideal and lowest values.

5.3.6.3 The Evaluation and Selection Process

The evaluation and selection process is performed to estimate the satisfaction between the COTS software alternatives and the evaluation criteria. Such estimation helps in selecting the fittest alternative. The process aims to collect, synthesize, and consulate the data in order to estimate each COTS alternatives and rank them based on their fitness scores. To do so, this process is carried out through two activities: i) data collection and filtering, and ii) decision making.

A. Data Collection and Filtering

This activity aims to collect the data of the CEC attributes that are related to the identified COTS software alternatives and estimate them based on the thresholds values (lowest values) which are defined in the yardstick. This process is essential in order to determine which of these COTS alternatives will be continued with more detailed evaluation and which of them will be eliminated. In this activity, the data collection is combined with the COTS alternatives filtering that aim to decrease the

number of the identified COTS software alternatives. This activity is more efficient in term of time and effort besides accelerating the evaluation process.

The data collection and COTS alternatives filtering run through several levels according to the data resources. As mentioned in section 5.3.4, there are four kinds of data collection techniques adapted which include the JAD, documents review, evaluation form, and software demonstration participation. Based on these techniques and their data resources, four levels of data collection and filtering are established. Each level includes a set of CEC attributes that share the same information resources and data gathering techniques (Figure 5.18). Appendix G describes the attributes of all the four levels.

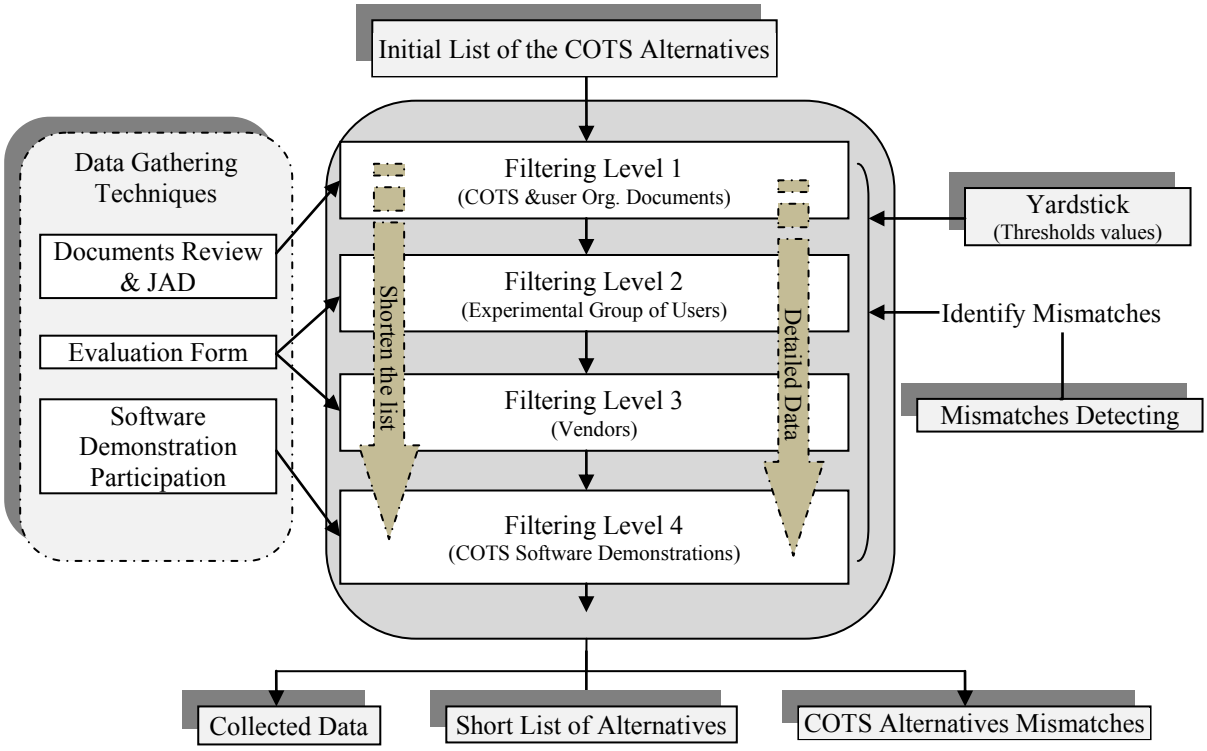


Figure 5.18. Data Collection and Filtering Activity

The first level of the data collection and filtering activity includes the set of CEC attributes that are related to the initial information. The information can be collected from the COTS software and user organization documents (first resource of information), as well as the stakeholders system. The collected information is used to estimate the COTS software alternatives by comparing it with the thresholds values defined in the yardstick. The COTS alternatives that do not satisfied the threshold values (lowest values) will be removed from the list. In this level, the documents review and JAD are specified as the data collection techniques.

In the second level of the data collection and filtering activity, the required information related to the set of attributes are requested from the experimental group users in the same domain, where each vendor provides a list of users that already used, tested, and customized the COTS software. The data are collected using the evaluation form technique. Figure 5.19 shows part of the evaluation form that is used to collect data related to some of the attributes in this level. The full evaluation form is included in Appendix F.

Please, answer the following questions by Yes (Y) or No (N)		**Yes**	**No**
1	Does the software product serialize its code and state? Which it can be transferred to a different machine, or stored for persistency.		
2	Does the software product store its state in a persistent manner for later recovery?		
3	If the software product provides any kind of mechanism or documentation that can be used by software tools or platforms for discovering and understanding its services, and for dynamically invoking them. (e.g. reflective mechanism, or UML or MOF (Meta-Object Facilities) descriptions of the software services and context.)		
4	Does the software product provide any mechanism/tool to prevent the data corruption (e.g. Backup facilities)?		
5	The user can be controlled its entire execution from the beginning to end (start, stop, and rerun the execution of the software product)		
6	The user can be controlled its interaction with the environment in which it is executing.		
7	The user can be switch control between the different functionalities provided by the software product while the software product is executing.		
8	Does the software product provide any mechanism/tool for recording and retrieving the user operations		

	data?					
9	Does the software product support or provide any tool or mechanism for creating a snapshot of the system's state to use for troubleshooting?					
10	Does the software product support or provide any tool or mechanism for detailed operational and functional reports?					
16	Does the software product avoid serious failure emergence	Always	Regularly	Sometime	Rarely	Never
		1	2	3	4	5
17	Does breakdown occur when users' task suspended	1	2	3	4	5
18	Does the vendor responds to user queries/faults and providing fixes to repair faults within the quoted time scales	1	2	3	4	5
19	Does the vendor provide regular (annually) releasing functional software upgrade	1	2	3	4	5
20	Does the vendor provide easy upgrade path.	1	2	3	4	5
21	Does the vendor support the software warranty services (ex. Warranty, guaranty, extended warranty/guaranty, maintenance services contract)	1	2	3	4	5

Figure 5.19. Evaluation Form for Level 2

In the third level, the related information of the attributes are requested from the vendors using the evaluation form technique. In the last level, when the number of the COTS software alternatives becomes small, the vendors of those COTS software products are invited to demonstrate their COTS software features in the presence of the evaluation team. The information that is gathered from the software demonstration is used to exclude those alternatives that are not achieving the threshold values of the attributes in this level.

At each level, the COTS alternatives that fail to satisfy the yardstick thresholds values will be excluded from the final list of the COTS software alternatives. Moreover, the mismatches between the COTS software alternative and the ideal values, as defined earlier in the yardstick, are identified for all COTS alternatives in the final list.

Finally, the main output of this task is the short-listed of the COTS software alternatives. However, the collected data of the COTS software alternatives and the set of identified mismatches are also included in the final list.

B. Decision Making

The decision making activity aims to synthesize and consulate the data from the previous activity and aggregate the weights of the CEC to make the decision of selecting the appropriate COTS alternative. In particular, this activity will receive the short list of the COTS alternatives; COTS software alternatives data, and the identified COTS software alternatives mismatches. As shown in Figure 5.20, the first task in this activity is to review the completeness and accuracy of information related to the CEC, yardstick values, collected data of each COTS alternatives, and COTS alternatives mismatches. In addition, the review is performed for other purposes. Firstly is to check whether the new functional requirements are raised especially after investigating the features of various COTS alternatives during the data collection and filtering activity. Secondly is to check if there are new COTS alternatives in the market. If so, the evaluation team will send back this new information as feedback to the previous process to conduct a second search according to the new requirements and collect the data about the new COTS alternatives.

After finishing the review and ensuring that the required information is complete, this information is aggregated in the form of meaningful numbers through the proposed decision making technique, which is described in detail in section 5.3.5.

This technique has an important role in synthesizing the identified mismatches and solution constraints (cost, effort, time, and risk) in order to compute the final score for the COTS alternative.

The evaluation team prepares the list that includes the sorted COTS alternatives based on their fitness scores. The COTS software alternative with the highest score is the most fitness software for the CEC and would be located on the top of the list, while the second highest alternative would be the second in the list and so on. The new list that contains the sorted COTS alternatives is referred as the COTS best-fit list. In addition, the relevant information that is needed in the next phases such as the information related to mismatches (e.g. mismatches solutions), the information related to the vendors (software supporting and upgrading), and COTS customization information are attached with the COTS best-fit list.

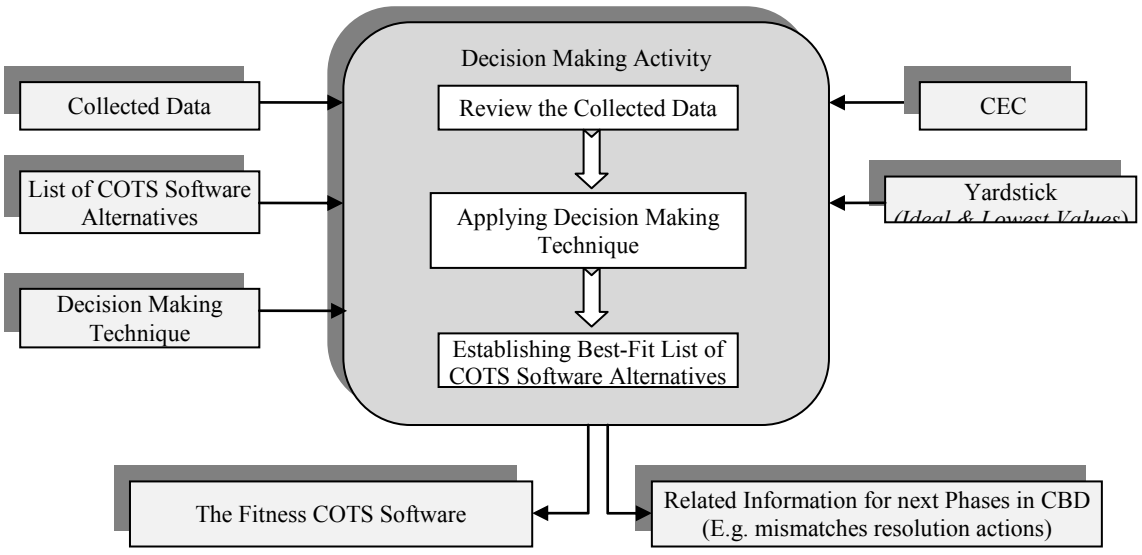


Figure 5.20. Decision Making Activity

The evaluation and selection process is done once the fittest COTS software is chosen and agreement with vendor is signed. The system development process will then move to adaption stage of the CBD by taking into consideration all information from the evaluation and selection process.

The data collection and filtering task as well as the proposed decision making technique of the task are applied throughout the development of the prototyping system tool (this is discussed in the section 6.3.1). This tool is used to conduct all the required calculations and comparisons that are needed during the decision making technique which helps to save the efforts and time for the COTS software evaluation and selection process. In addition, the use of this tool helps to lessen the roles of the evaluation team. These roles are summarized as follows:

1. Determining the threshold (ideal & lowest) values of each attributes in the CEC in order to use for filtering the COTS software alternatives and identifying the mismatches.
2. Assigning the judgments in pairwise comparisons during the weighting step.
3. Determining the COTS mismatches resolution actions and estimating their constraints (cost, time, effort, and risk).

5.4 Summary

In this chapter, the research problems are addressed by proposing COTS-ESF. This framework was developed based on the evaluation theory that provides six evaluation components for any evaluation process. These components were

integrated and consolidated in three main processes: planning, preparation, and evaluation and selection. The COTS-ESF was developed to select the fitness COTS software through coupling the CEC that were established and proposed as the common evaluation criteria for evaluating COTS software with the decision making technique that provides systematic and well-defined process to estimate the fitness of the COTS software alternatives.

CHAPTER SIX
FRAMEWORK EVALUATION

6.1 Introduction

After its construction, the proposed COTS-ESF, as described in the previous chapter, has been implemented to provide better understanding. This chapter presents the findings of the framework evaluation, including the implementation of a prototype system tool based on real-life case study. The process of evaluating COTS-ESF was carried out through the verification and validation stages.

The discussion in this chapter begins by describing the activities in the verification and the validation stages and the findings discussion.

6.2 Verification by Expert Review

The verification was performed to ensure that the main components in the proposed COTS-ESF, such as CEC and decision making technique, were built correctly. It was carried out through experts review method using Delphi technique. The Delphi technique was conducted through three rounds revisions as shown in Figure 6.1.

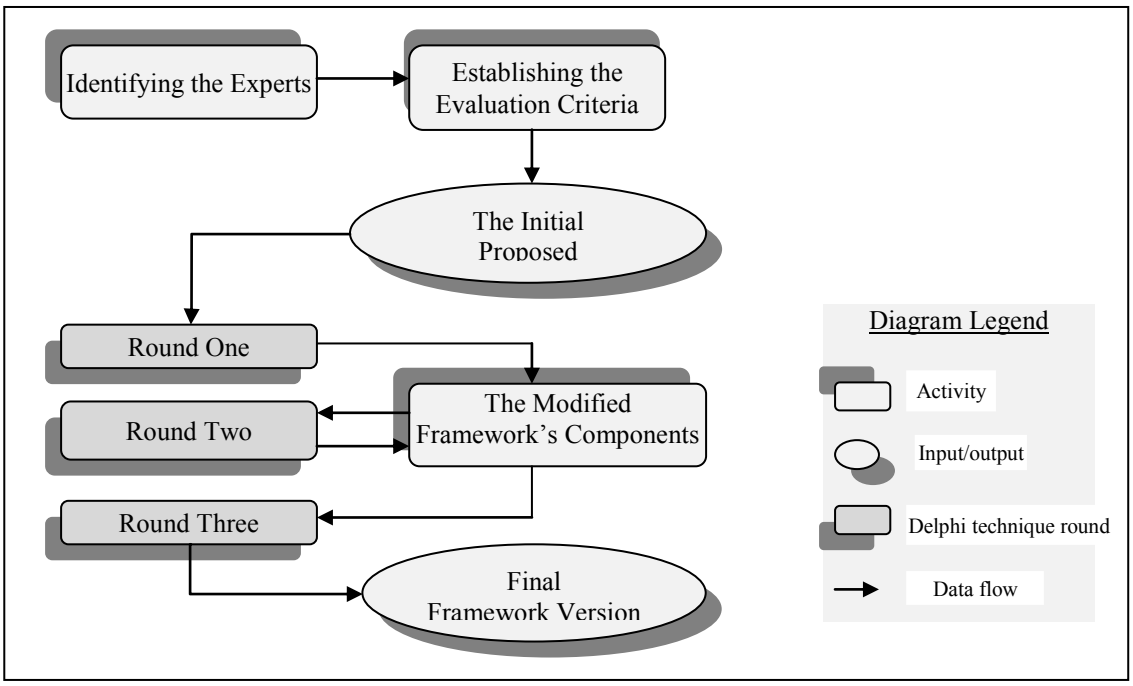


Figure 6.1. Proposed COTS-ESF Verification Process Using Delphi Technique

The verification process starts by identifying the potential experts, as discussed in chapter 3, related to the COTS software evaluation domain. In fact, 35 experts have been identified from different countries and were contacted through e-mail. However, only 15 of them accepted to review and evaluate the proposed framework. Unfortunately, after the first round, three of those experts withdrew from continuing the verification process. Therefore, only 12 experts participated in reviewing and evaluating the proposed framework throughout the verification stage. The participated experts comprised of eight professional systems developers and four academic researchers from different environments and countries. This number of experts is considered adequate based on the recommendation by Hallowell and Gambatese (2010) and Rowe and Wright (1999) where at least the three experts are still accepted to apply the Delphi technique and conducted experts review.

The identified developers represent different environments from different countries which are: USA (1), Qatar (1), Saudi Arabia (3), Malaysia (1), and Jordan (2). All of them have at least 3 to 5 years experience in developing, evaluating, and purchasing the systems. The academic researchers also represent different perspectives from UK (1) and Malaysia (3). All of them are PhD holders (professor and associate professor) with many years of experience as a faculty member and consultant in the industry. They also have many publications in the area related to the COTS software evaluation.

Four criteria were used to verify the estimation of the proposed COTS-ESF by the experts. These criteria include comprehensiveness, understandability, accuracy, and coherence (Behkamal et al., 2009; Kunda, 2002; Moody et al., 2003). The following are the descriptions of those criteria.

1) **Comprehensiveness**: This criterion looks for the inclusion of all required processes, tasks, techniques, and characteristics for evaluating the COTS software. It also indicates the coverage of all related viewpoints.
2) **Understandability**: This criterion is to evaluate the models from the standpoint of software engineering. Based on this criterion, the model structure and components should be clear, usefulness, appropriate for audience, ease to use, ease to implement, and unambiguous. Ambiguity occurs when there is an incorrect interpretation among the model components and their relationships.

3) **Accuracy**: The accuracy of the model is determined based on the decision support, expected results, usable results, cost-effective, and the adequate number of characteristics, sub-characteristics, and attributes that should be decomposed to achieve precise evaluation.
4) **Coherence**: This criterion indicates the extent of compatibility, well-organized, and internally consistency of the model's structure and components.

These criteria were used to form the questionnaire related to the verification process. The questionnaire was designed to verify the main processes, tasks, techniques, the proposed decision making technique, and the CEC. Therefore, it consists of three parts which are: i) the main processes and tasks; ii) the proposed decision making; and iii) the CEC. The questions were adapted from the previous studies such as Kunda (2002), Behkamal et al (2009), and Kitchenham and Pickard (1998). To answer the questions related to these parts, there are three scales that were used to reflect the feedback from the respondents, which are: "Yes without modifications"; "Yes with modifications"; or "No". The questionnaire is found in Appendix J.

Three rounds were required to complete the proposed COTS-ESF verification. The following sections discuss the results of each round and the required modifications.

6.2.1 Results of Round One

After identifying, contacting, and getting acceptance from the experts, the first round of the verification stage began by sending the questionnaire with the proposed

COTS-ESF through e-mail. This round aims to give the experts an opportunity to study the framework's components carefully and be aware with the questionnaire. The time limit for sending their feedback is about four weeks. The feedback was analyzed once received. Table 6.1 shows the important information related to this round.

Table 6.1
Round One Information Summarization

Name	Round One
Aim	To carefully study the proposed COTS-ESF and get the feedback
Inputs	Questionnaire & Identified Experts Emails
Reviewers	Expert 1, Expert 2, …, Expert 12
Contact Method	By E-mail
Duration	15 - 40 days

The following parts present the answers of the three parts of the verification questions and the related suggestions for each part.

1. **Answers and suggestions of the part one**

The experts were asked to answer 10 questions in this part in order to verify the proposed processes, tasks, and techniques in the COTS-ESF in term of its' comprehensive, understandable, accuracy, and coherence. In this respect, three options were used to answer these questions which are: "Yes without modifications", "Yes with modifications", and "No". The answers that selected by the experts are presented as the following.

A. The experts answers related to the part one

In the first question about whether the proposed processes, tasks, and techniques are perceived usefulness, the majority of the experts answered "Yes without

modifications", while the remaining of them (three experts) chose "Yes with modifications". In term of the clarity and understandability of the framework's processes and tasks, eight of the experts answered "Yes without modifications" while four of them said that they need further explanation. More details in Table 6.2.

Table 6.2

The Part One Answers of Verification Questionnaire

CRITERIA	Yes without modifications		Yes with modifications		No	
	Freq	%	Freq	%	Freq	%
Perceived usefulness	9	75	3	25	0	0
Clear and understandable	8	66.7	4	33.3	0	0
Coverage the required COTS selection tasks	10	83.4	1	8.3	1	8.3
Appropriate for the COTS selection task	10	83.4	1	8.3	1	8.3
The framework's presentation	4	33.3	8	66.7	0	0
Easy to implement	8	66.7	4	33.3	0	0
Allow users participation	8	66.7	4	33.3	0	0
Integrating data collection and COTS filtering tasks is correct and cost-effectiveness	8	66.7	4	33.3	0	0
The used techniques are adequate and sufficient	9	75	2	16.7	1	8.3
Provide expected results and completed information	10	83.4	1	8.3	1	8.3

B. The experts suggestions

Amongst the answers of the experts related to this part, there are several suggestions have been provided by the experts to improve the proposed COTS-ESF. These suggestions are concluded in the following points:

- For those who do not have enough experience to evaluate and select COTS software, the experts stressed that the proposed process and tasks require being simple to understand and ease to implement where each processes and

task has clear inputs, steps, and outputs. In addition, it is worthy to show the order of the evaluation processes in the implementation phase where planning process should be labeled as “First Process: Planning”, preparation Process labeled as “Second Process: Preparation”, and so on.

- Since the important of user participation in COTS software evaluation and selection process, it is necessary to determine in which activity the users will participate and their roles. For the same purpose, increasing the participation of the stockholders such as the software user, and the experts in the field.

- Regarding the proposed techniques, four experts suggested to provide further explanation about the JAD technique. In particularly, it is required to determine the number of JAD sessions, specify the tasks in each session, and recognize the roles of JAD members.

- To improve the data collection and filtering task, four experts suggested that this task will be more clear and efficient when each level of COTS software filtering is matched with appropriate data collection technique such as the level one can collect the data through the document review technique.

2. Answers and suggestions of the part two

This part consists of seven questions in order to verify the proposed decision making technique. Therefore, three choices were used to answer the questions which are “Yes without modifications”, “Yes with modifications”, and “No”.

The following points present the answers and the experts' suggestions related to this part's questions.

A. The experts' answers related to the part two

At looking to the Table 6.3, most of the experts selected "Yes without modifications" to answer the questions in this part while the remaining of them answered "Yes with modifications" and suggested some of modifications to improve the proposed decision making technique.

Table 6.3

The Experts' Answers Related to the Second Part of the Questionnaire

CRITERIA	Yes without modifications		Yes with modifications		No	
	Freq	%	Freq	%	Freq	%
Correctness of the proposed technique	10	83.3	2	16.7	0	0
Accurate result satisfaction	8	66.7	4	33.3	0	0
Completeness of the proposed technique	9	75	3	25	0	0
Consistent and well-organized structure	10	83.3	2	16.7	0	0
Easy to implement by using the software tool	10	83.3	2	16.7	0	0
The used equations are valid and sufficient	11	91.7	1	8.3	0	0
Using mismatch's constraints (time, cost, and risk) in calculating mismatches level support selecting the fitness COTS software	11	91.7	1	8.3	0	0

B. The experts suggestions

In this part, some experts provided a few suggestions to make the proposed decision making technique understandable and easy to implement. These suggestions are presented in the following points.

- In term of the results satisfaction, the proposed technique should provide required explanation to the unsuccessful COTS products, why these products

were unsuccessful. This can be addressed by providing the information related the each COTS software mismatches for each vendor such as the mismatch level, potential solution, and the constraints.

- Since the proposed decision making technique relies on several equations, steps, and comparisons it is important to make it easy to understand and implement, thus an example should be provided about how to apply the proposed steps and equations in this technique.

3. Experts' answers and suggestions of the part three

In this part of the questionnaire, the proposed CEC was verified through four questions where three choices were provided to answer these questions which are "Yes without modifications", "Yes with modifications", and "No". The following parts present the experts' answers and their suggestions.

A. The experts' answers related to the part three

The experts were asked four questions related to the CEC model. When the experts were asked to determine whether the CEC is enough to evaluate COTS software, (50%) of them selected "Yes with modifications" while the rest of them answered "Yes without modification". Regarding the other questions, the answers of the most experts were "Yes without modifications" while few of them suggested some of modifications to improve the CEC. For more details see Table 6.4.

Table 6.4

The Experts' Answers Related to the Part Three

CRITERIA	Yes without modification		Yes with modifications		No	
	Freq	%	Freq	%	Freq	%
CEC enough to evaluate and select COTS	5	41.7	6	50	1	8.3
CEC clear and easy to understand	10	83.3	2	16.7	0	0
CEC adequate to achieve precise evaluation	7	58.3	5	41.7	0	0
CEC's structure consistent and compatible with standard model's structure (e.g. ISO9126)	10	83.3	2	16.7	0	0

B. The experts' suggestions

With regards to the suggestions that provided by the experts in the third part of the questionnaire, the answers of the experts were summarized in the following suggestions:

- Provide clear definitions for the characteristics, sub-characteristics, and attributes in CEC to make it more understandable and it's better to refer to the appropriate quality standard such as ISO 9126.

- The safety characteristic is very critical and important for most of the software product especially for COTS software. Therefore, it should be added to the CEC and it is more related to the architecture category.

- In order to make the CEC more accurate, understandable, and easy to implement it is required to define clearly the metrics and data type for each attribute in the CEC.

- To mitigate the complexity of CEC, some unrequired attributes and metrics need to be removed and some other need to integrate together such as the attributes under the vendor reputation.

- Reorganize the quality and architectural categories and their characteristics to be more understandable and coherence like moving the reusability characteristic in quality category is more suitable to be in the architectural category.

- Change the name of the organization category to be more related to the characteristics inside this category, such as "Operational Environment category". In addition, the names of the characteristics and sub-characteristics require improving to meaningful names in order to be more understandable (such as supportability under architectural category will confuse with supportability under vendor category). In this regard, the experts suggested using an international standards terminology such as ISO standard.

- To make some measurement attributes easy to measure, the experts were agreed that it's better to change some of complex measures to simple measures. For example, using likert scale in some attributes because of the difficulties to get the real data especially the data related the COTS software, like the data related to the response time and incorrect operational failure.

Based on the results of this round, the required modifications for COTS-ESF were incorporated and used as inputs in the second round.

6.2.2 Results of Round Two

After completing the data collection and analysis, the results in round one were prepared as input for the next round. The second round aimed to modify COTS-ESF based on the required modifications that were made known from the first round. Table 6.5 summarizes the required modifications that were needed to modify the proposed COTS-ESF.

Table 6.5

The Required Modifications to Improve the COTS-ESF

The Level	The Required Modifications
Processes, tasks, and techniques Level (***Part One***)	• The proposed processes and tasks require being simple to understand and implement where each processes and task should has clear inputs, steps, and outputs. In addition, provide a certain label for each process to show its' order in the implementation phase in order to avoid confusing about which processes is coming first. • Determine in which activity the users will participate and their roles. For the same purpose, to increase the reliability of the evaluation process, it's better to increase the participation of the stockholders such as the experts in the field. • Provide further explanation about the JAD technique where the number of sessions, tasks and roles should be specified. • To improve the data collection and filtering task, the experts suggested that this task will be more efficient when each level of filtering is matched with appropriate data collection technique.
Decision Making Technique Level (***Part Two***)	• In terms of the decision making results satisfaction, the proposed technique should provide required explanation to the unsuccessful COTS products, why these products were unsuccessful through providing information related to the each COTS mismatches as a report that can be accessed by the vendor to check his product's weaknesses. • The proposed decision making technique will be ease to understand if an example is provided about how to apply the proposed steps and equations in this technique.

The CEC Level (***Part Three***)	• Provide clear definitions for the characteristics, sub-characteristics, and attributes in CEC to make it more understandable. • Add safety in the used characteristics under the architecture category. • Define the metrics and data type for each attribute in CEC. • Remove some unrequired attributes and metrics that increase the complexity of the CEC such as the attributes under the vendor reputation. • Reorganize the quality and architectural categories and their characteristics to be more understandable and coherence like moving the reusability characteristic in the quality category is more suitable to be in the architectural category. • Change the name of the organization category to be "Operational Environment category". In addition, the names of the characteristics and sub-characteristics require improving to meaningful names in order to be more understandable (such as supportability under architectural category will confuse with supportability under vendor category). In this regard, the experts suggested using an international standards terminology such as ISO standard. • To make some measurement attributes more easy to answer, the experts agreed to change some of complex measures to be simple measures such as using likert scale in some attributes because of the difficulties to get the actual data such as in the response time and incorrect operational failure.

Thus, based on these modifications, the COTS-ESF was modified as shown in Table 6.6. As for the inputs, the modified COTS-ESF was used as inputs in the third round. The details information about this round is presented in Table 6.6.

Table 6.6

Round Two: Information Summarization

Name	**Round Two**
Aim	To improve the proposed COTS-ESF based on experts suggestions from round one
Inputs	Results from round one (experts' suggestions) and Proposed COTS-ESF
The Taken Actions	
The processes, tasks, and techniques level	
• Each process and their tasks in the COTS-ESF were provided by further explanation and diagram to clearly show the inputs, outputs, and their steps. In addition, the main processes in the framework were renamed to show their order in implementation as following: (**First Process**: planning), (**Second Process**: Preparation), (**Third Process**: Evaluation & Selection). Refer to section 5.3.6. • The COTS-ESF was modified to allow the users to participate in several tasks which are: the functional requirements gathering, yardstick defining, CEC weighting, and decision making. In addition, the COTS-ESF also was modified to allow forming the evaluation team to join several kinds of stockholders which are: experts, manager, decision	

maker, technical, and users in order to increase the reliability of the evaluation process. Refer to section 5.3.6.1.

- The JAD technique was further explained and its benefits, potential members, and their roles were also provided. Refer to section 5.3.4.
- The task of data collection and filtering was improved through matching between each level of data collection and filtering with suitable data collection technique as following: first level matches with document review and JAD techniques; the second and third levels match with the evaluation form, and the fourth level matches with software demonstration participation. Refer to section 5.3.6.3.

The Decision Making Level

- The decision making technique was modified to provide the information related to the COTS mismatches as report in the final result. In this report, set of information are provided which are: the attributes that have the mismatches, the mismatch level, and the potential solution. By this report the vendors can identify the weakness point of their products and how to improve them. Refer to section 5.3.5.
- To make the decision making technique ease to understand, an example was provided for each steps in order to provide a guideline for using the equations and the steps in the technique. Refer to section 5.3.5.

The CEC Level

- Provide accurate definitions for each criterion in CEC according to ISO 9126 and previous models such as Bertoa and Valecillo (2002), and Kalaimagal and Srinivasan (2010/a). Full description about CEC in Appendix D.
- Other actions have been made as following:

Category	**Characteristic**	**Sub-characteristic**	**Attribute**	**Type**
Quality characteristics	Functionality	Compliance (-) Accuracy (r) to correctness	Certification (-) Standardization (-)	
	Efficiency	Resource behavior	I/O devices utilization (-)	
		Time behavior	Response time	Time (c) to level
			Capacity	
			Throughput	
	Maintainability	Replaceability (-)	Replacement ease level (-)	
		Stability (+)	Stability level (+)	Ratio (c) to level
	Usability	Operability	Complexity ratio (-)	
		Understandability	Help system (r) to the quality of help system	
			User documentation (r) to quality of user documentation	
Domain char. (r) to domain quality characteristics	Popularity	Internet discussion	Bloggers/ email (-)	
Architecture char. (r) to architectural quality characteristics	Portability	Replaceability (+)	Replacement ease level (+)	Level
	Supportability (r) to operational support	Health monitoring (-)	Health models (-)	
	Safety in use (+)	Risk of software (+)	Risk level (+)	Level
User organization char. (r) to operational environment	Current system platform char. (r) to system platform char.	Hardware platform	I/O sub-system (r) to data transfer system	
	Software	People (r) to	Training course (-)	

characteristics	development environment	developers		
	Staff expertise and culture (r) to culture	Organizational culture	(Polices (-), business roles and procedures (-), policies and roles (+)	
	Financial issue	Acquisition costs	Maintenance costs (-)	
		Further development costs	Mismatches and upgrading costs (r) to adapting cost	
Vendor organization characteristics.	Reputation	Management quality (-)	Operations management (-)	
			Customer satisfaction (-)	
			Risk management (-)	
		Innovativeness (-)	Innovative product (-)	
	Stability	Track record	The first developed date (-)	
			Growth the organization (-)	
		Employees	Organizing (-)	
	Supportability	Delivery	Confirmation date of delivery (-)	
		User training (r) to quality of training	User documentation (-)	
			Training courses (r) to quality of training courses	

Note: (+) means has been added new criterion. (-) means criterion has been removed.
(r) means has been rename to. (c) means has been changed to

Based on the round two results, the required modifications were conducted to improve the COTS-ESF. The main outputs of this round were the required modifications and the improved COTS-ESF.

6.2.3 Results of Round Three

The required modifications and the improved COTS-ESF established in round two were sent to the expert as a new round in order to get their approval and acceptance. In this third round, the e-mail was used to communicate with the experts regarding the required modifications and improvement of the COTS-ESF. As a result from this round, the experts agreed with the processes, tasks, techniques, and CEC of the COTS-ESF. Table 6.7 shows the main information related to this round.

Table 6.7

Summary of Round Three

Name	Round Three
Aim	To get the acceptance from the experts about the COTS-ESF after the modifications
Inputs	Required modifications + the COTS-ESF after modifications
Contact Method	By E-mail
Duration	8 - 21 days
Results	The COTS-ESF was accepted by experts to evaluate the COTS software

6.3 Validation stage

The objective of the validation process is to evaluate the effectiveness and applicability of the COTS-ESF. To do so, there are two approach have been used to carry out the validation stage, which are: case study and yardstick validation.

6.3.1 Validation by Case Study

This section presents the results of the two case studies conducted in Malaysia and Jordan. These case studies describe the implementation of COTS-ESF in evaluating and selecting the appropriate COTS software. The aim is to validate the COTS-ESF and to show its applicability and added benefits. The section begins by describing the prototype system support tool that was used to facilitate the application of the COTS-ESF. Next is to describe the two case studies and discuss their results.

6.3.1.1 Decision Making-Prototyping Tool (DM-PT)

The DM-PT is a prototype tool aims to help evaluating and selecting the COTS software in a systematic way according to the proposed decision making technique. The tool is to automate and accelerate the process of performing many calculations

and comparisons such as the CEC weighting, and also dealing with complicated and tedious data required in the proposed decision making technique. In addition, it helps in preventing the occurrence of errors during the execution process especially when the final decision of selecting the fittest COTS software depends on the accuracy of these computations.

The DM-PT was built using the ASP.Net technology and the Visual Basic.Net (VB.Net) programming language, and Microsoft SQL Server 2008. The main components of this tool, as shown in Figure 6.2, are:

1) User-interface that is used by users to interact with other components.
2) Computational module that used to perform required calculations and comparisons for the previous formulas such as formula 5.3, 5.4, 5.8, 5.9, 5.11, 5.12, and 5. 13.
3) Database storage that uses to store information related to the evaluation criteria, the attributes (measures), the COTS software alternatives, and the COTS mismatches.

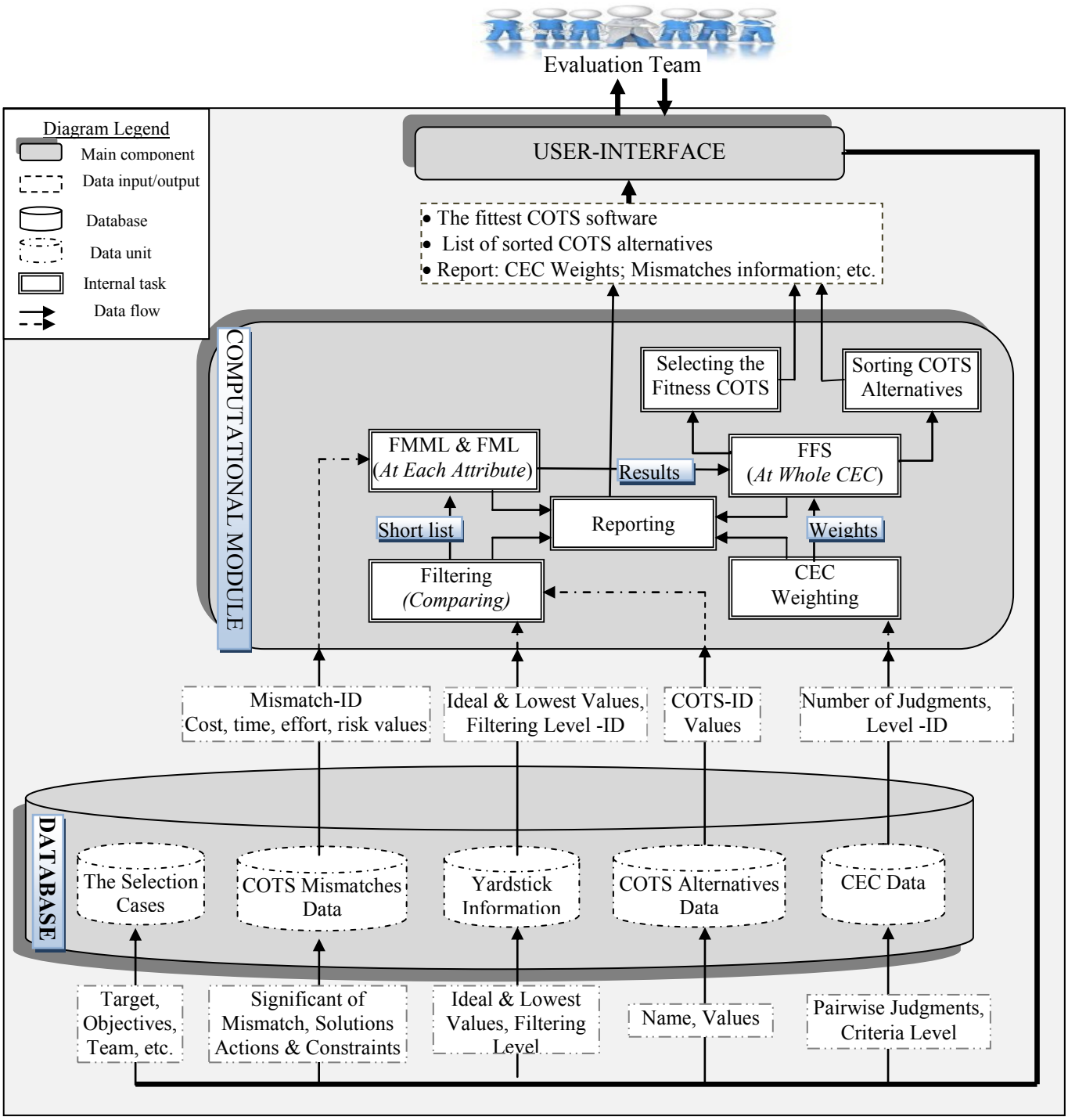


Figure 6.2. The Main Components and Their Interactions of DM-PT

The snapshots are provided to highlight some of the steps in the DM-PT. The main menu is the first display in the DM-PT as demonstrates in Figure 6.3, which includes several choices, which are: 1) Case Selection; 2) Defining Yardstick; 3) CEC weighting; 4) COTS software alternatives; 5) Data collection and filtering; 6) COTS

Mismatches handling; 7) Fitness score calculation; 8)The fitness COTS software; and 9) Reports.

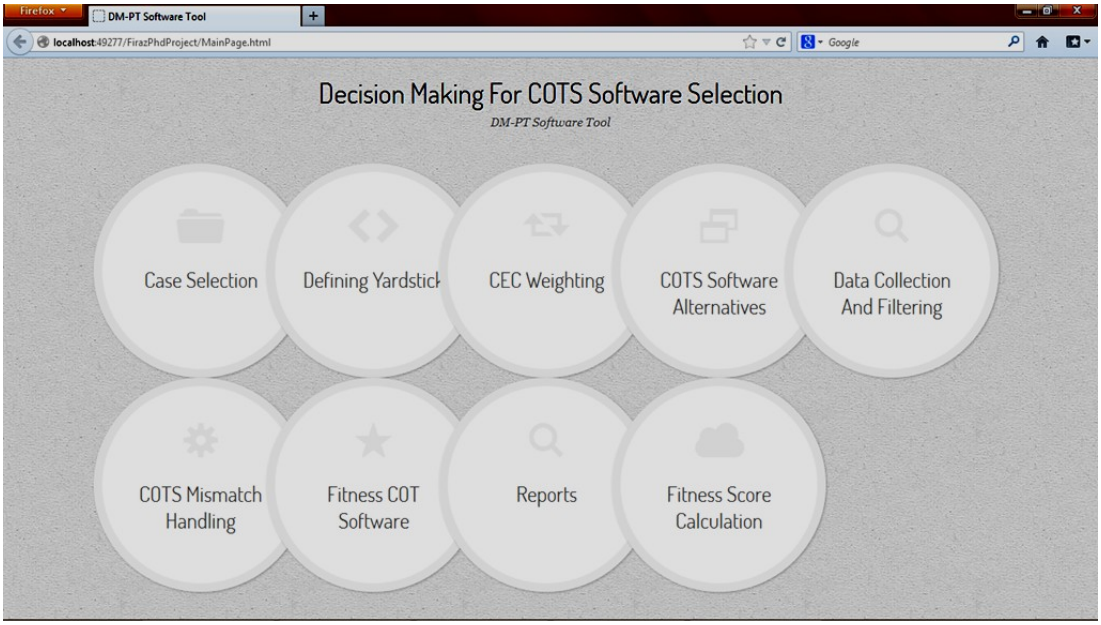


Figure 6.3. Snapshot of the Main Screen of DM-PT

The system starts with users entering the project WBS information in the planning process by choosing the ***Case Selection.*** The ***Defining Yardstick*** screen enables the evaluation team to key in the thresholds (ideal and lowest values) of the yardstick which can be executed in the preparation process. The data collection and filtering activities in the evaluation and selection process is executed through the ***Data Collection and Filtering*** screen, which provides four levels of the activities based on the source of data as explained in section 5.3.4. In the ***CEC weighting*** screen, the weights are calculated and assigned to the evaluation criteria in the CEC based on the pairwise judgments inputs from the evaluation team. Selecting the menu choice of ***COTS mismatches handling*** enables the evaluation team to enter the potential

resolution action and estimate the values of its constraints (costs, effort, time, and risks) for those COTS mismatches that have significant impact on the fitness of COTS software. The Final Fitness Score (FFS) computation for each of the COTS alternative is carried out through the choice of ***fitness score calculation*** in the main menu. The main output of this tool is displayed by selecting the ***fitness COTS software*** that shows the appropriate or fittest COTS software (the highest FFS). The ***Reports*** choice presents the final list of the COTS alternatives and displays a set of details information such as the Final Mismatching Level (FMML) at each attribute and the potential solutions.

As shown in Figure 6.4, the DM-PT handles most of the framework processes and activities. Each member of the evaluation team has specific roles at each process in using this tool. For example, at the preparation process, the evaluators define the ideal and lowest values in the yardstick and at the evaluation and selection process, the evaluators key-in their judgments in the pairwise comparisons to calculate the CEC criteria weightage. As the same time, some of the required information about the COTS mismatches (e.g. solution, effort, and risk) is keyed in. Then the final score for each COTS alternative is calculated in order to determine the fittest alternative.

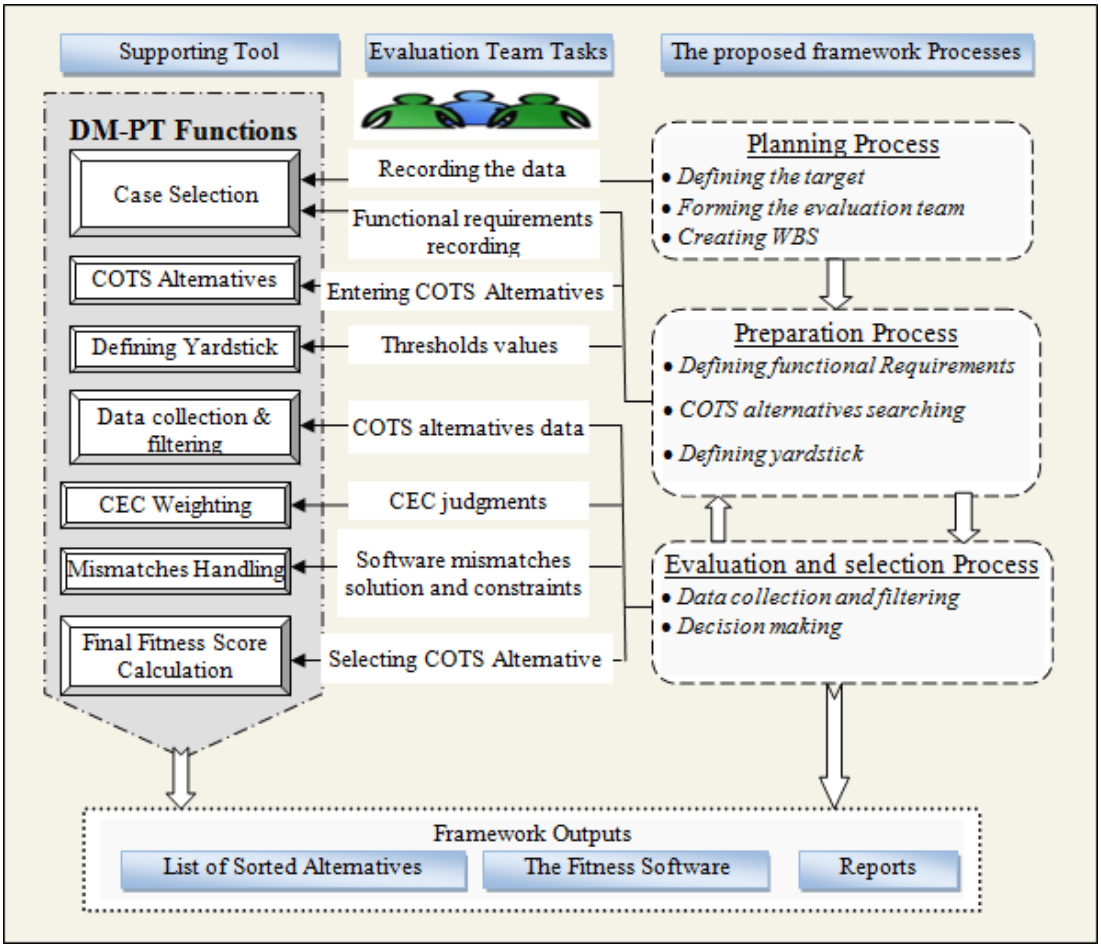


Figure 6.4. Processes Alignment with DM-PT

The detailed descriptions of the activities conducted during the validation process for both organizations A and B are presented in the following sections.

6.3.1.2 Case Study One: Selecting the Appropriate Security System (Anti-Virus Software)

6.3.1.2.1 Profile of Organization A

Organization A is one of the public universities in Malaysia. It was established to primarily develop and promote management education in the country. Its academic

programs are especially geared towards providing a broad spectrum of academic knowledge and intellectual skills in many areas such as management, accounting, economics, information technology, and public administration. Currently, there are approximately 29,000 students and 2,800 employees at this organization. This organization consists of three main colleges: College of Business (COB), College of Arts and Sciences (CAS), College of Law, Government, and International Studies (COLGIS). It also comprises of thirteen departments, four institutes, nine centers and two units.

Recently, the organization needed to integrate new security software (anti-virus) to protect its system. Thus, the Computer Center of this organization was given the responsibility to search and select appropriate anti-virus software from the software market. After contacting and discussing with the Computer Center manager and staff in charge, the proposed framework was used to evaluate and select the fittest anti-virus software.

Anti-virus software components are now common in any computer system. It is developed to protect the systems and computers against malware and malicious programs such as viruses, worms, Trojans, and spyware. It is also designed to infect and take control of the target system. In the recent years, there are a lot of anti-virus software products with different features and capabilities in the market.

Once an agreement and acceptance has been achieved, the manager of the computer centre mandated the head of the in charge unit to apply the COTS-ESF for selecting

and evaluating the most appropriate anti-virus software. The unit team were given a week to study and understand the descriptions of the COTS-ESF. After that, a meeting was held with the head to clarify any misunderstanding or ambiguity of the COTS-ESF.

The next sections describe the details of the COTS-ESF related processes in evaluating and selecting the anti-virus software in this organization.

6.3.1.2.2 Planning Process

By referring to section 5.3.6.1, this process consists of three activities: i) defining the evaluation target, ii) forming the evaluation team, and iii) creating the WBS. These activities are described in detail as follows.

A. Defining the Evaluation Target

In this activity, the manager, unit head, and researcher held a discussion to define and delimit the project target. The main target of this project was to determine the appropriate anti-virus software that fit the organization resources and at the same time help the organization to achieve high productivity. The anti-virus software can be considered as a type of COTS software since it is developed and supported by the third party (vendors) to be sold to users. Since there are many anti-virus software in the market, the scope of the targeted software has to be determined. In this case, the scope identified refers to the ability of the software to provide the cloud computing technology support. This process is summarized in Table 6.8.

Table 6.8

Defining the Project Target

Defining the Target	
Aims	To identify and delimit the project target.
Inputs	Security problem in the organization systems
Outputs	Well-defined project target
Duration	1 day
Roles	Computer center manager, and the in-charge unit head
Used technique	JAD technique

B. Forming the Evaluation Team

Other roles and tasks that are required to carry out the project were determined once process A is completed. This is shown in Table 6.9.

Table 6.9

Different Roles in the Project

Role	Tasks
Project leader	Planning and managing the project such as managing the time and budget of the project, estimating CEC weights, and defining yardstick thresholds
Evaluator	Defining the functional requirements, CEC weights, and yardstick thresholds, searching the anti-virus software alternatives, collecting data and handling the mismatches (identifying the solutions and constraints)
Domain Expert	Defining functional requirements, estimating the CEC weights and defining the yardstick thresholds.
User	Defining functional requirements and estimating the CEC weights

After determining the above-mentioned roles and contacting the potential stakeholders, five members were enrolled in the process of acquiring suitable anti-

virus software. These members are represented by M1, M2, M3, M4, and M5 as described in Table 6.10.

Table 6.10

Evaluation Team Members

Member	Roles	Skills	Experience (years)
M1	Project Leader	Manager	13
M2	Domain Expert	Systems Security	20
M3	Evaluator	Technical	12
M4	Evaluator	Technical	6
M5	User	Non technical	6

The process of forming the evaluation team is summarized in Table 6.11.

Table 6.11

Forming the Evaluation Team

Forming the Evaluation Team	
Aims	To determine the members of the evaluation team together with their tasks and responsibilities in carrying out the project.
Inputs	The project target and the stakeholders.
Outputs	Evaluation team members and their tasks
Duration	2 days
Roles	Computer center manager, the in-charge unit head
Technique	JAD technique

C. Creating the Project WBS

The third activity of the planning process is to establish the project WBS. This requires a meeting between the Computer Center manager and the evaluation team to determine the main issues related to the project such as budget, time, and main

objectives and constraints. In this context, the DM-PT can support and facilitate the creation of the project WBS by enabling the evaluation team to enter and store all required information. Figure 6.5 displays the snapshot of the data entered during the meeting related to the project WBS creation.

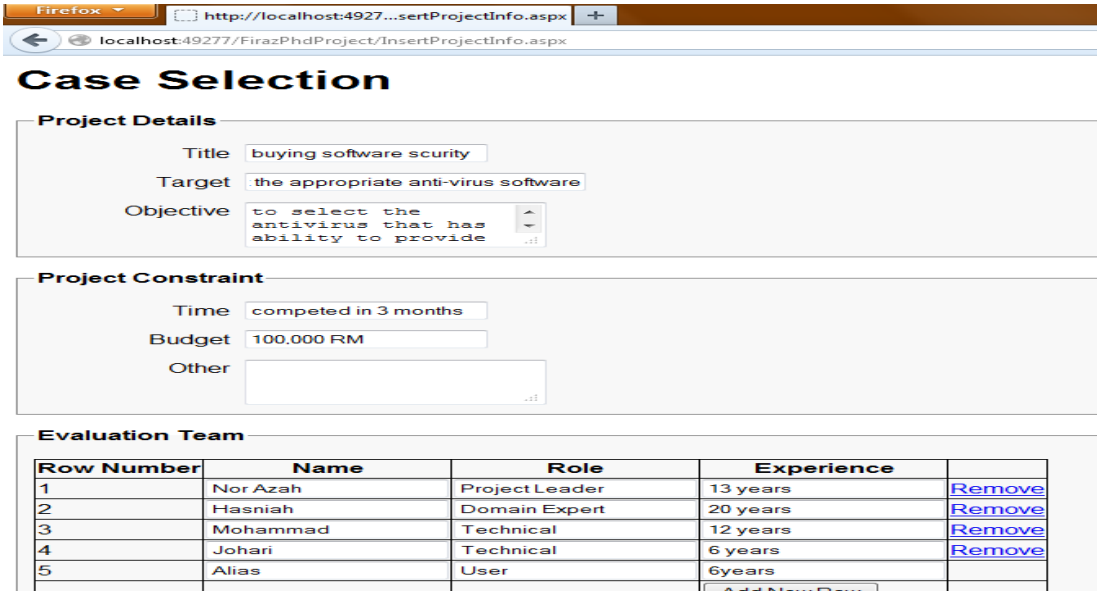


Figure 6.5. The Snapshot of the WBS

The main information required in this activity is presented in Table 6.12.

Table 6.12
Creating Project WBS

	Creating Project WBS
Aims	To establish the WBS of the project which include target objectives, project constraints (budget and time), etc.
Inputs	The project target, evaluation team, responsible unit head and manager.
Outputs	Project WBS
Duration	1 day
Roles	Computer center manager, in-charge unit head, and evaluation team.
Technique	JAD technique

It is important to mention that the DM-PT was set up during this process and its demonstration was conducted in the presence of the evaluation team to make sure they can use it easily.

6.3.1.2.3 Preparation Process

In this process, the required information was collected for further evaluation (refer to section 5.3.6.2) through three activities: defining functional requirements, COTS alternatives searching, and defining the yardstick.

A. Defining the Functional Requirements

The functional requirements were identified during this activity by the evaluation team members (M2, M3, M4, and M5). Using the JAD technique, the first meeting with users and stakeholder was held to elicit the functional requirements. The related documents in the organization were studied and analyzed during the brainstorming session. Then the second JAD meeting was conducted with other team members to discuss and analyze these requirements. The identified required functional requirements are summarized in Table 6.13.

As a result of the trade-off between the functional requirements identification and the next activity, the software alternatives searching (Figure 5.15), the functional requirement number 5 (Table 6.13) was modified to include “All reports are exportable and can be configured to be sent automatically by e-mail”. This modification was done due to the appearance of the desired feature during the searching activity.

Table 6.13

Identified Functional Requirements

Id	Identified Functions	Description
1	Highly effective spam filtering	A guaranteed filtering mode to assure effective filtering, optimize the bandwidth and the company resources and reduce the time dedicated to spam management.
2	Management delegation	The anti-virus software should enable administrators to delegate or split administration tasks between other privileged users, assigning them the computers to which they can access.
3	Email quarantine management	A quarantine area that stores all suspected mail with spam or potentially containing viruses. This ensures that the end-client's servers are not saturated by spam, saving resources and capacity.
4	Monitoring	The anti-virus allows user to run real-time or scheduled analysis of network and monitor the results at any time.
5	Reporting	The software can also generate configurable reports with protection status and detection activity and graphs. ***All reports are exportable and can be configured to be sent by email automatically.***
6	Peer-to-peer automatic updates	Workstations and servers update and upgrade their protection from the nearest desktop optimizing bandwidth consumption. If the update package is not found in the LAN, the workstation will get it from internet. Update frequency can be configured, and carried out on demand or by groups.
7	Profile-based protection	Offering different protection profiles to assign to different users, groups of users or domains according to the organization needs.

Then, the results were entered and recorded to the DM-PT software tool as shown in Figure 6.6.

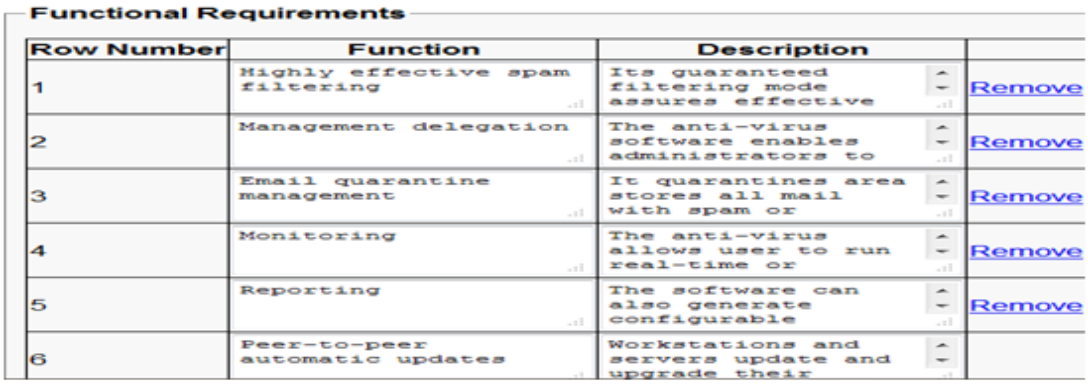


Figure 6.6. The Functional Requirement Stored in the DM-PT

The final outputs of this activity were set of functional requirements to be used in the next activity. Table 6.14 summarizes the main information related to this activity.

Table 6.14

Key Information Related to Defining Functional Requirements

	Defining the functional requirements
Aims	To identify the functional requirements.
Inputs	Project target, Evaluation team, stakeholders, system documentation, and feedback from searching activity
Outputs	Functional requirements (*refer to Table 6.9*)
Duration	3 days
Roles	Expert, evaluator, and user
Used technique	JAD technique and document review

B. The COTS Searching Activity

The main purpose of this activity was to identify the potential anti-virus software based on the functional requirements identified in the previous activity. The evaluators used several searching sources, such as internet (Google and Yahoo engines) and specialized websites that provide a list of anti-virus software (i.e. www.toptenantivirus.info). In addition, market searching was also used by going through the list of identified vendors. To narrow the searching scope, the evaluation team used the following option in the desired anti-virus software choice, "it should use the cloud technology or cloud-based anti-virus". The initial result was a list of 15 COTS software products. The information of the related COTS alternatives collected from the websites and other sources were then keyed-in to the DM-PT (Figure 6.7).

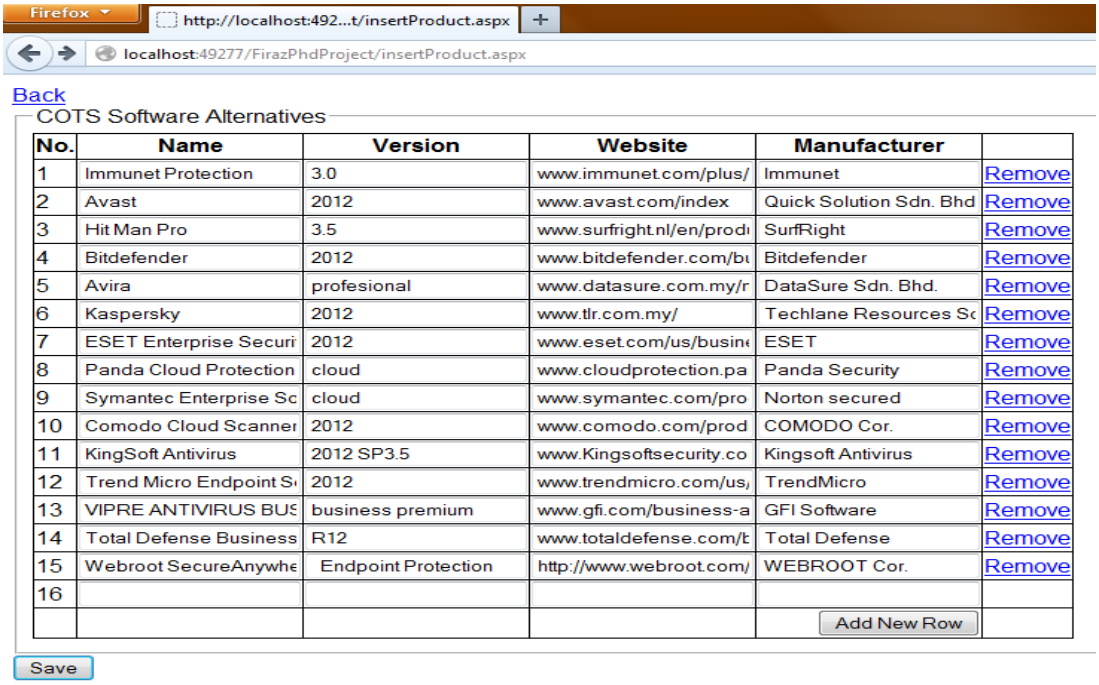


Figure 6.7. Snapshot Screen Displays: Entering the Anti-Virus Alternatives

Table 6.15 sums up the COTS searching activities.

Table 6.15

The COTS Software Searching Activity

	COTS Software Searching
Aims	To identify the potential anti-virus software that meets most of the functional requirements.
Inputs	Functional requirements, searching sources (internet and market), and evaluation team.
Outputs	List of potential anti-virus software (refer to Figure 6.7).
Duration	5 days
Team members	Evaluators
Used technique	Search engines (e.g. Google), local market (list of local agents)

C. Defining the Yardstick

At this stage, another meeting was held by the evaluation team to determine the ideal and lowest values of the yardstick. These values addressed various kinds of information such as those related to security system domain, financial issues, technical issues, and system usability. Therefore, in this activity, the domain expert (M2), project leader (M1), evaluator (M3), and user (M5) were invited to attend this meeting. The project leader (M1) was responsible for the finance and legal issues, while the evaluator (M3) for the technical issues. Two meetings were required, each for completing the defining and reviewing the yardstick thresholds values. During these meeting the DM-PT was used to enter the yardstick thresholds values. The yardstick attributes were also defined and explained in the DM-PT as well as the data type for each attribute. This was done to help and facilitate the evaluation team to determine the thresholds values and save it in a systematic way. Figure 6.8 displays the snapshot screen during the entering a group of yardstick thresholds values of this project. Full information about the yardstick thresholds is in Appendix H.

Save

ID	Attribute Name	Data_Type_Name	Ideal Value	Lowest Val
1	Coverage	perecentage	90	70
2	Excess	perecentage	90	70
3	Service implementation coverage	perecentage	90	70
4	Memory utilization	integer	1	2
5	Disk utilization	integer	1	2
6	COTS price	integer	50000	70000
7	Delivery (installation) cost	integer	0	2000
8	Infrastructure upgrading cost	integer	1000	5000
9	Mismatching/adapting cost	integer	1000	3000
10	integration costs	integer	1000	5000
11	COTS testing cost	integer	0	1500
13	Hardware dependency	bolean (1/0)	0	0
14	Software dependency	bolean (1/0)	0	1
15	installation Document	bolean (1/0)	1	1
16	Deployment document	bolean (1/0)	1	1
17	Data compatibility	bolean (1/0)	1	1
18	Version compatibility	bolean (1/0)	1	0
19	Training	bolean (1/0)	1	1
20	current operating system	bolean (1/0)	1	1
21	current middleware (e.g. CORBA standard)	bolean (1/0)	1	1
22	communication applications	bolean (1/0)	1	1
23	processing unit performance	Five-likerit scale	1	3
24	Memory system	Five-likerit scale	1	3
25	Data transfer system	Five-likerit scale	1	3
26	The quality level of user documentation	Five-likerit scale	1	4
27	development process (tasks, roles, processes)	Five-likerit scale	1	2
28	Supplementary process (standards, guidelines, checklists, templates)	Five-likerit scale	1	3
29	Development tools (Integration and configurations tools, install and scripts tools, debugging and testing tools, etc)	Five-likerit scale	1	3
30	Developers' Skills/knowledge	Five-likerit scale	1	3
31	expertise	Five-likerit scale	1	4
32	Users' knowledge/skills	Five-likerit scale	1	4
33	expectations	Five-likerit scale	1	2
34	behavior (General operating norms, Cooperative, Interaction)	Five-likerit scale	1	3
35	Symbols	Five-likerit scale	1	4
36	Language	Five-likerit scale	1	3
37	Policies and roles	Five-likerit scale	1	4

Figure 6.8. Snapshot Screen Displays: Defining a Group of the Yardstick Thresholds

Also, Table 6.16 shows summary of defining the yardstick activity.

Table 6.16

Defining the Yardstick Activity

	Defining the yardstick thresholds
Aims	To determine the thresholds values (ideal & lowest values) of the yardstick.
Inputs	The yardstick attributes and their data types (which provided by DM-PT) and evaluation team (M1, M2, M3, and M5).
Outputs	Thresholds values for the attributes in the yardstick (refer to Figure 6.8 and Appendix H).
Duration	2 days
Team members	Expert, project leader, evaluator, and user
Used technique	JAD technique

6.3.1.2.4 The Evaluation and Selection Process

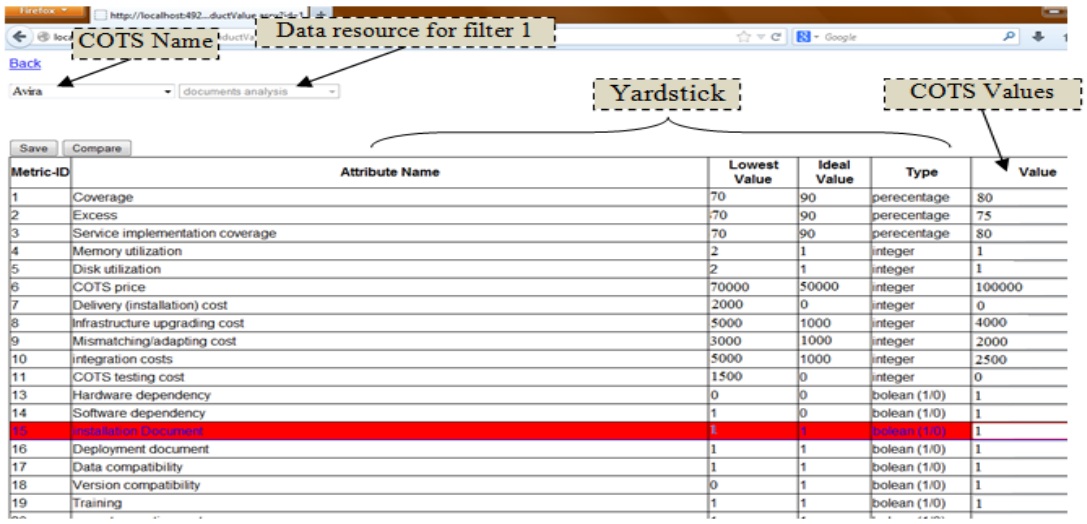
In the evaluation and selection process, the data about the anti-virus software alternatives was gathered and synthesized in order to select the fittest anti-virus software. This process started by collecting and filtering activity to collect the data and short the list of alternatives, and then selecting the fittest one by carrying out the decision making activity (refer to section 5.3.6.3). More descriptions about these activities are in the following sections.

A. Data collection and Filtering Activity

In the data collection and filtering activity, the aim is to collect the related information of the antivirus alternatives and decrease their list. This activity is divided into four levels based on the number of the data sources. At each level the alternative that did not meet the yardstick values was eliminated from the list. The first level of data collection and filtering of the COTS-ESF starts by studying and analyzing the documents of the user organization and COTS products. These documents provide the basic data. The evaluation team (evaluator and user) also used the document attached to the anti-virus products or those available on the websites (Figure 6.9). The collected information was entered to the DM-PT that included all the attributes of the four levels of data collection and filtering activity (refer to 5.3.6.3).

Figure 6.9. Snapshots from an Anti-Virus Website

The DM-PT automatically filtered out all the products that did not match the thresholds values in that level. Figure 6.10 shows the snapshoot screen from the data collection and filtering in level one.



Metric-ID	Attribute Name	Lowest Value	Ideal Value	Type	Value
1	Coverage	70	90	perecentage	80
2	Excess	70	90	perecentage	75
3	Service implementation coverage	70	90	perecentage	80
4	Memory utilization	2	1	integer	1
5	Disk utilization	2	1	integer	1
6	COTS price	70000	50000	integer	100000
7	Delivery (installation) cost	2000	0	integer	0
8	Infrastructure upgrading cost	5000	1000	integer	4000
9	Mismatching/adapting cost	3000	1000	integer	2000
10	integration costs	5000	1000	integer	2500
11	COTS testing cost	1500	0	integer	0
13	Hardware dependency	0	0	bolean (1/0)	1
14	Software dependency	1	0	bolean (1/0)	1
15	Installation Document	1	1	bolean (1/0)	1
16	Deployment document	1	1	bolean (1/0)	1
17	Data compatibility	1	1	bolean (1/0)	1
18	Version compatibility	0	1	bolean (1/0)	1
19	Training	1	1	bolean (1/0)	1

Figure 6.10. The First Level of the Data Collection and Filtering

After each level, the DM-PT produced a short report that indicates the status of the software, either pass or fail. This is portrayed in Figure 6.11.

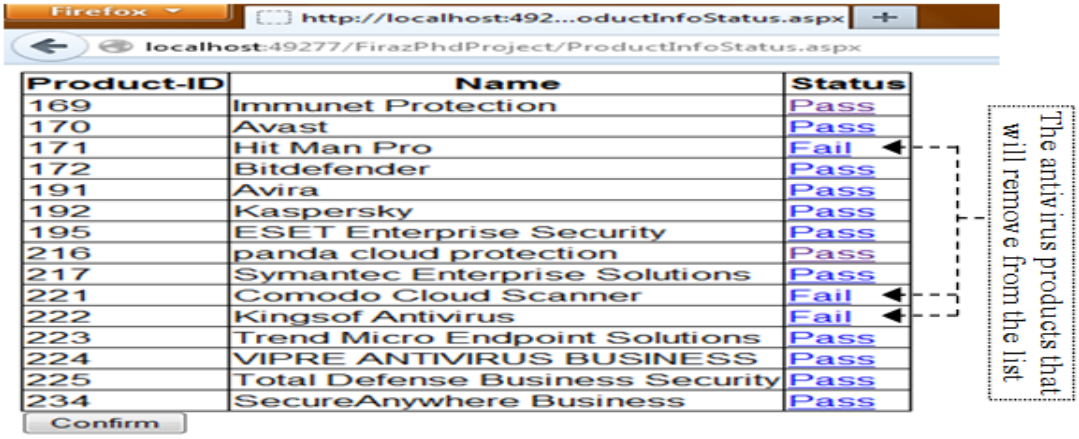


Figure 6.11. The Data Collection and Filtering Result of Level One

From the first level, three products were eliminated from the list. In the second level, the evaluation team used the evaluation form to collect the data from other users of the product (Appendix F). Some of those users were known to this organization and the others were provided by the vendors as the client reference. Twelve evaluation forms were sent to those users and they were given two weeks to response. Once received, the response was entered to the DM-PT for filtering purposes. Those products that did not match the yardstick thresholds values were eliminated from the list. As a result, only ten anti-virus products remained. In the third level, any unavailable information from the products documents and other product users is requested from the vendors. The request was made using the evaluation form that includes all the attributes related to the vendor, as shown in Appendix F. The evaluation form was also used to collect information about the vendors of those products that passed the filtering process from level two. In this case, the evaluation

forms were sent to ten vendors who developed the anti-virus products. This number of vendors resulted from the previous level. The vendors were requested to fill these forms within two weeks. This data was then entered to the DM-PT. The result showed that only 6 products passed the filtering process. As for the final or forth level, some of the information were collected only during the software product demonstration. The demonstration was conducted by those vendors that remained in the list in the presence of the evaluation team. Some of the vendors performed their demonstration through their websites whereby customers can use their trial version for a limited period to test and access the features of their products. At this level only four anti-virus products passed the filtering process (Figure 6.12).

Figure 6.12. The Forth Level Filtering Result

The products that fulfilled the yardstick thresholds values and met the functional requirements needed to be further evaluated before selecting the most appropriate

anti-virus product. This was done in the next activity. Table 6.17 presents the main information related to this activity.

Table 6.17

Data Collection and Filtering Activity

Data Collection and Filtering Activity	
Aims	To collect the data about the software products and short list these products.
Inputs	The list of software products from the searching activity, yardstick thresholds values that provided by DM-PT, and evaluation team.
Outputs	Short List anti-virus software for further evaluation (refer to Figure 6.12).
Duration	45 days
Roles	Evaluator and user
Used technique	Evaluation form, software product demonstration attending , JAD and document review techniques

B. Decision Making Activity

The decision making activity aimed to further evaluate the list of the anti-virus products from the previous activity in order to select the fittest anti-virus software for this organization. To do so, the decision making technique proposed in this framework was applied. The technique involved tasks: CEC weighting and COTS alternatives scoring (refer to section 5.3.6.3):

i) CEC Weighting

The CEC weighting was performed to assign weightage to the evaluation criteria in order to distinguish them according to their importance. Another meeting between the domain expert, project leader, user, and evaluator was conducted to contribute their judgments (from 1 to 9) through several pairwise comparisons at each level in the CEC. The evaluation criteria in the CEC are provided by the DM-PT as a set of

three levels pairwise matrixes. More details about assigning the weights, pairwise matrixes, and consistency were discussed in section 5.3.5.1. The DM-PT was keep running so that the evaluation team can directly keyed-in their judgments and check the consistency of these judgments at each matrix. Figure 6.13 displays a snapshot screen of the evaluation team judgments related to the pairwise matrix for the first level of the CEC (the main categories).

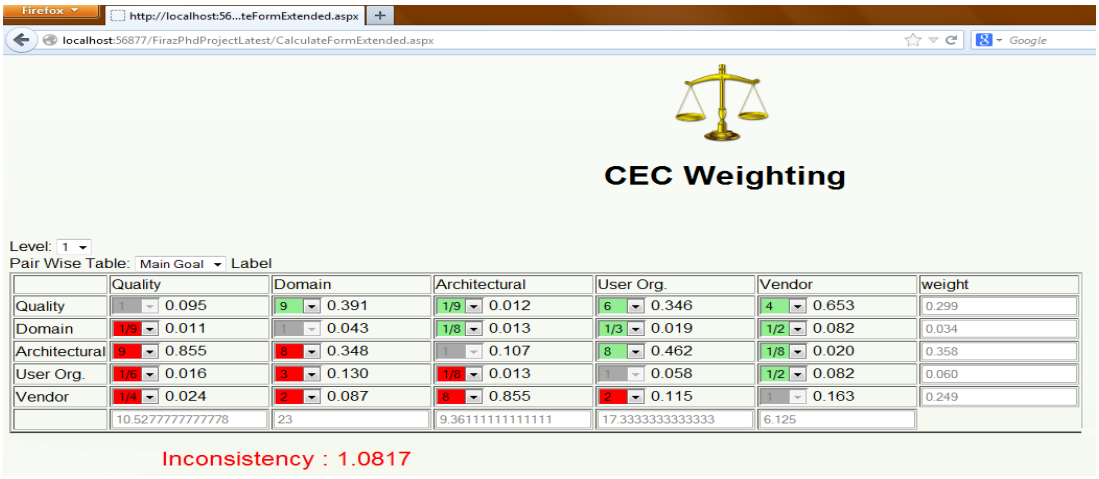


Figure 6.13. Snapshot Screen: The First Level of the Pairwise Matrix

As shown in the previous Figure, the matrix was inconsistent because the consistency ratio (CR) was equal to 1.0817. Since the CR was more than 0.10, the evaluation team needed to review their judgments in the matrix so that the consistency can be improved. The improved result (CR is equal 0.085 < 0.10) is shown in Figure 6.14.

Figure 6.14. The Pairwise Matrix after Reviewing the Judgments

The weights of all the criteria in the CEC are displayed in Appendix H. Table 6.18 presents the main information related to this weighting task.

Table 6.18

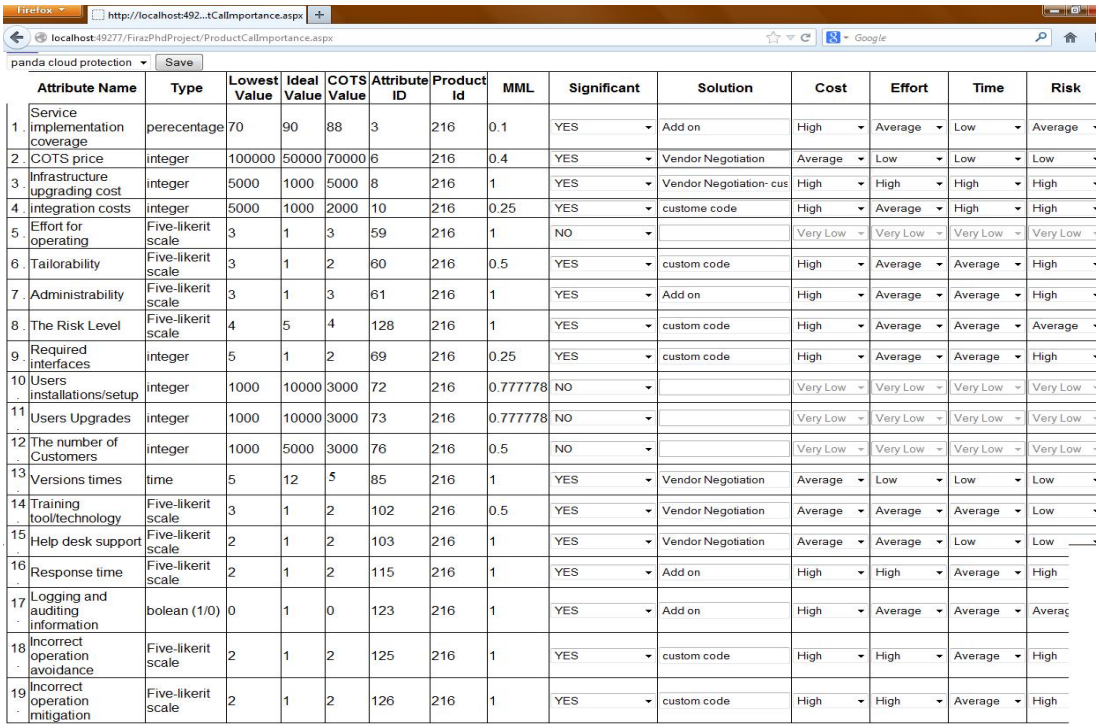
Main Information Related to the CEC Weighting Task

CEC weighting task	
Aims	To assign weights to the criteria in the CEC.
Inputs	The CEC criteria (which provided by DM-PT) and evaluation team.
Outputs	CEC weights (refer to Figure 6.14 and Appendix H).
Duration	1 day
Team members	Expert, project leader, evaluator, and user
Used technique	JAD technique to conduct the meetings

ii) COTS Alternatives Scoring

The COTS alternatives scoring were to calculate the final scores for each COTS alternative against the CEC. This scoring task was conducted by handling the mismatches, potential solution and constraints (cost, effort, time, and risk) (refer to section 5.3.5.2). The DM-PT played a vital role in this task by performing all the

required calculations. The evaluation team had a limited role in selecting the detected COTS mismatch that is significant or not because the all computations were conducted systematically by the DM-PT. The team was only required to enter the potential solutions for those significant COTS mismatches and estimate their constraints (cost, effort, time, and risk). Figure 6.15 displays a snapshot of the anti-virus software mismatches as well as their solutions and constraints.



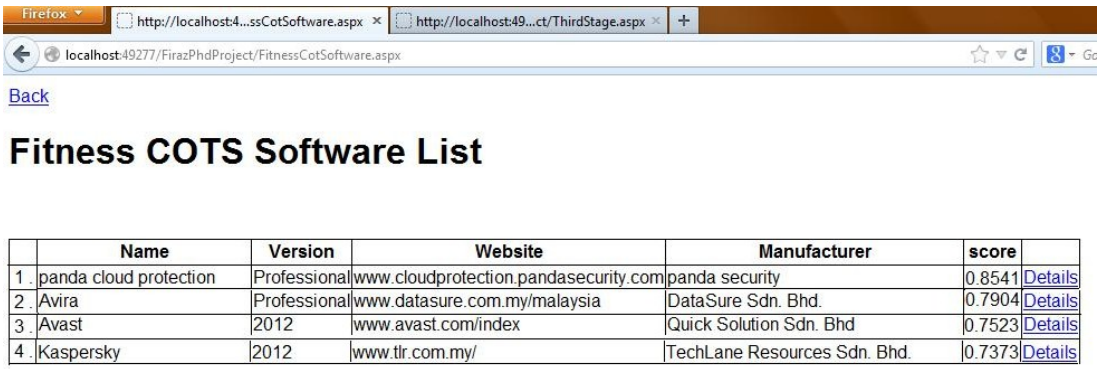
	Attribute Name	Type	Lowest Value	Ideal Value	COTS Value	Attribute ID	Product Id	MML	Significant	Solution	Cost	Effort	Time	Risk
1	Service implementation coverage	perecentage	70	90	88	3	216	0.1	YES	Add on	High	Average	Low	Average
2	COTS price	integer	100000	50000	70000	6	216	0.4	YES	Vendor Negotiation	Average	Low	Low	Low
3	Infrastructure upgrading cost	integer	5000	1000	5000	8	216	1	YES	Vendor Negotiation- cus	High	High	High	High
4	integration costs	integer	5000	1000	2000	10	216	0.25	YES	custome code	High	Average	High	High
5	Effort for operating	Five-likerit scale	3	1	3	59	216	1	NO		Very Low	Very Low	Very Low	Very Low
6	Tailorability	Five-likerit scale	3	1	2	60	216	0.5	YES	custom code	High	Average	Average	High
7	Administrability	Five-likerit scale	3	1	3	61	216	1	YES	Add on	High	Average	Average	High
8	The Risk Level	Five-likerit scale	4	5	4	128	216	1	YES	custom code	High	Average	Average	Average
9	Required interfaces	integer	5	1	2	69	216	0.25	YES	custom code	High	Average	Average	High
10	Users installations/setup	integer	1000	10000	3000	72	216	0.777778	NO		Very Low	Very Low	Very Low	Very Low
11	Users Upgrades	integer	1000	10000	3000	73	216	0.777778	NO		Very Low	Very Low	Very Low	Very Low
12	The number of Customers	integer	1000	5000	3000	76	216	0.5	NO		Very Low	Very Low	Very Low	Very Low
13	Versions times	time	5	12	5	85	216	1	YES	Vendor Negotiation	Average	Low	Low	Low
14	Training tool/technology	Five-likerit scale	3	1	2	102	216	0.5	YES	Vendor Negotiation	Average	Average	Average	Low
15	Help desk support	Five-likerit scale	2	1	2	103	216	1	YES	Vendor Negotiation	Average	Average	Low	Low
16	Response time	Five-likerit scale	2	1	2	115	216	1	YES	Add on	High	High	Average	High
17	Logging and auditing information	bolean (1/0)	0	1	0	123	216	1	YES	Add on	High	Average	Average	Averag
18	Incorrect operation avoidance	Five-likerit scale	2	1	2	125	216	1	YES	custom code	High	High	Average	High
19	Incorrect operation mitigation	Five-likerit scale	2	1	2	126	216	1	YES	custom code	High	High	Average	High

Figure 6.15. Product 1 Mismatches and Their Solutions

After determining the required information for each product mismatches in the list, the DM-PT computed the individual anti-virus product Final Fitness Score (FFS). Then only the fittest anti-virus product was selected. This is shown in Figure 6.16.

Figure 6.16. Snapshot Screen: the Fitness Anti-Virus Product

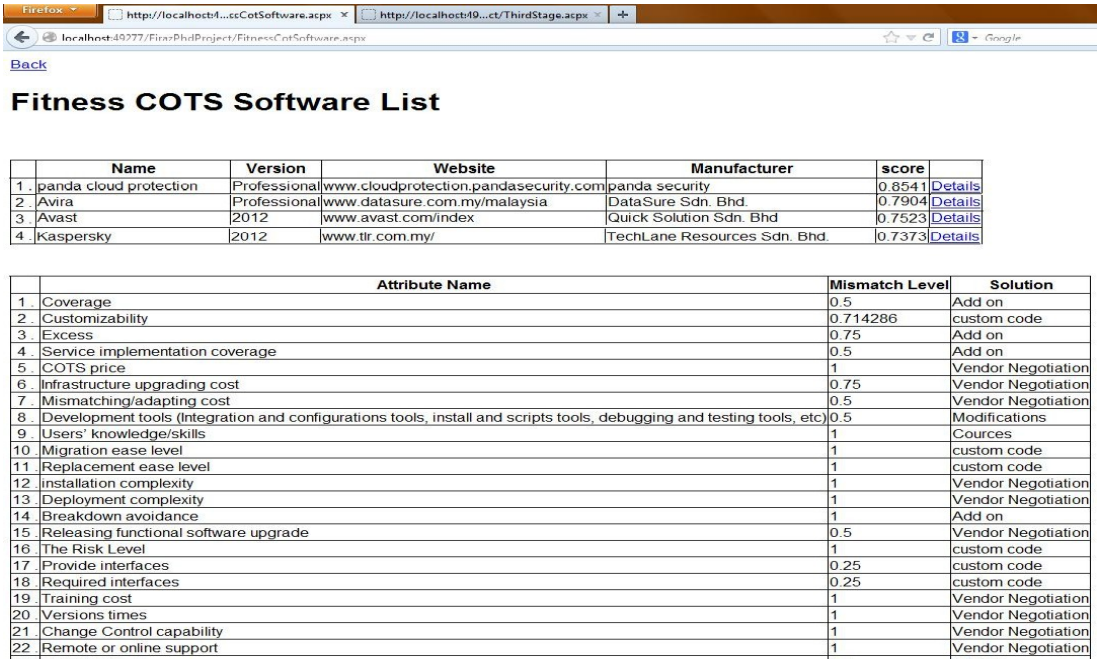
Figure 6.17 presents the sorted software products list according to their fitness to the evaluation criteria.



	Name	Version	Website	Manufacturer	score	
1 .	panda cloud protection	Professional	www.cloudprotection.pandasecurity.com	panda security	0.8541	Details
2 .	Avira	Professional	www.datasure.com.my/malaysia	DataSure Sdn. Bhd.	0.7904	Details
3 .	Avast	2012	www.avast.com/index	Quick Solution Sdn. Bhd	0.7523	Details
4 .	Kaspersky	2012	www.tlr.com.my/	TechLane Resources Sdn. Bhd.	0.7373	Details

Figure 6.17. Snapshoot Screen: the Sorted List of Products

Figure 6.18 displays an example of the product mismatches report, their levels and solutions. The result of the fittest anti-virus product and its report were used in the next phase where it is integrated to the organization system. In addition, these reports were sent to the vendors as feedback for them to improve their product in the next versions. The reports for each product in the list are shown in Appendix H.



Back

Fitness COTS Software List

	Name	Version	Website	Manufacturer	score	
1 .	panda cloud protection	Professional	www.cloudprotection.pandasecurity.com	panda security	0.8541	Details
2 .	Avira	Professional	www.datasure.com.my/malaysia	DataSure Sdn. Bhd.	0.7904	Details
3 .	Avast	2012	www.avast.com/index	Quick Solution Sdn. Bhd	0.7523	Details
4 .	Kaspersky	2012	www.tlr.com.my/	TechLane Resources Sdn. Bhd.	0.7373	Details

	Attribute Name	Mismatch Level	Solution
1 .	Coverage	0.5	Add on
2 .	Customizability	0.714286	custom code
3 .	Excess	0.75	Add on
4 .	Service implementation coverage	0.5	Add on
5 .	COTS price	1	Vendor Negotiation
6 .	Infrastructure upgrading cost	0.75	Vendor Negotiation
7 .	Mismatching/adapting cost	0.5	Vendor Negotiation
8 .	Development tools (Integration and configurations tools, install and scripts tools, debugging and testing tools, etc)	0.5	Modifications
9 .	Users' knowledge/skills	1	Cources
10 .	Migration ease level	1	custom code
11 .	Replacement ease level	1	custom code
12 .	installation complexity	1	Vendor Negotiation
13 .	Deployment complexity	1	Vendor Negotiation
14 .	Breakdown avoidance	1	Add on
15 .	Releasing functional software upgrade	0.5	Vendor Negotiation
16 .	The Risk Level	1	custom code
17 .	Provide interfaces	0.25	custom code
18 .	Required interfaces	0.25	custom code
19 .	Training cost	1	Vendor Negotiation
20 .	Versions times	1	Vendor Negotiation
21 .	Change Control capability	1	Vendor Negotiation
22 .	Remote or online support	1	Vendor Negotiation

Figure 6.18. Snapshot Screen from the Product 2 Report

The summary of this task is presented in Table 6.19.

Table 6.19

Decision Making Activity

	Decision Making
Aims	To select the fittest anti-virus products and generate reports about the main results (e.g. mismatches, solution) related to the software products in the list.
Inputs	The short list of software products from the data collection and filtering, the detected mismatches that provided by DM-PT, and evaluation team.
Outputs	The fitness of the anti-virus software (refer to Figure 6.16), list of sorted software products (refer to Figure 6.17) and the reports for each software products (refer to Appendix H).
Duration	1 day
Roles	Project leader, Evaluator and User
Used technique	Decision making technique supported by DM-PT

At the end of this activity, a meeting was held by the evaluation team to discuss the results and prepare the final report to be submitted to the Computer Center manager. The results were obtained by the project leader and the head unit in the Computer Center.

6.3.1.3 Case Study Two: Selecting the Appropriate Student Management Information System (SMIS)

6.3.1.3.1 The Profile of Organization B

Organization B was founded in 1999 as one of the important educational organizations in Jordan. It is a comprehensive public university in a self-contained campus. The student populations represent nearly every Governorate in Jordan. This organization is heavily involved in strategic planning to achieve its goals, vision, and mission and to deliver programs of study that meet national and international standards and accreditations measures. Over the past years, this organization has grown to eight colleges offering bachelor degree programs in natural and environmental sciences, business, nursing, education, humanities, IT and engineering. It also comprises of two Deanships: the Deanship of Student Affairs and the Deanship of Scientific Research. It has nine scientific centers that are heavily engaged in research and development projects to serve local and national communities improve students' life quality and deliver study programs.

In order to manage the student information and support the high quality decision making related to the students affairs, this organization chose to buy the student

management information system (SMIS). It is a software application for educational establishments to manage the students' information. This system addresses the needs of Admissions, Registration, Financial Aid, Student Accounts, Academic History, Graduation, Student Housing and Student Immunizations. Direct access to information supports improved decision making in areas such as admissions, enrollment management and student support. It varies in size, scope and capability, from packages that are implemented in relatively small organizations to cover student records alone, to enterprise-wide solutions that aim to cover almost all aspects of running large multi-campus organizations with significant local responsibility.

The process of selecting the SMIS application was done using the COTS-ESF in this research. After contacting with the director of the Computer and Information Technology Center who was responsible for developing and purchasing the required software applications, a meeting was established to discuss the used of COTS-ESF for evaluating and selecting the SMIS applications. As a result of this discussion, the director accepted the recommendation of using COTS-ESF in facilitating his decision making. Eventually, the full description of COTS-ESF was sent to him. He was given six days to study and understand the framework. After those six days, the project started applying COTS-ESF to select the most appropriate SMIS application for the organization. Detailed descriptions of the framework processes are presented through the following sections.

6.3.1.3.2 Planning Process

There are three activities involved in the planning process, which include 1) defining and delimiting the project target; 2) forming evaluation team; and 3) establishing the WBS of the project. The detailed of these activities are described as the following.

A. Defining the Evaluation Target

The SMIS is a kind of COTS software since it is developed, provided, and supported by vendors and is available to the general public in multiple copies without source code. Therefore, it met the target of the framework. In this activity, a meeting was held between the Center's director and the various in-charge section heads to delimit the target of the project. From the meeting, it was decided that the project was carried out to integrate a comprehensive information system that deals with students' information and records such as the academic and financial records. The system should also support and facilitate in making high quality decisions within a short time and less efforts. This activity is presented in Table 6.20.

Table 6.20

Defining the Evaluation Target

Defining the project target	
Aims	To define and delimit the target of the project.
Inputs	Current problem relate to the students information in the organization system
Outputs	Well-defined project target
Duration	1 day
Roles	Computer and IT center director and other competent members
Used technique	JAD technique

B. Forming the Evaluation Team

After finishing this first activity, the director of the Center started to identify the evaluation team members. The COTS-ESF in this activity recommends that the evaluation team should include a manager or leader, an expert in the related domain, an end-user, and a technician. The main roles of the team members in conducting the project are depicted in Table 6.21.

Table 6.21

Different Roles in the Project

Role	Tasks
Project leader	Leading the project planning and execution, estimating the CEC weights, defining the yardstick thresholds, and reviewing the results of each stage in the project.
Evaluator	Represents the technical side who is responsible for defining the functional requirements, CEC weights, defining yardstick thresholds, searching the software alternatives, collecting data and handling the mismatches (identifying the solutions and constraints)
Domain Expert	Estimating the CEC weights and defining the functional requirements and the yardstick thresholds.
User	Defining the functional requirements and estimating the CEC weights

Once determining the main roles, the director continued to contact various internal and external stakeholders of the Center in order to identify qualified evaluation team members. Finally, six members were selected to join in the team. The role and skill of each member is listed in Table 6.22.

Table 6.22

The Evaluation Team Members

Member	Role	Skill	Experience (years)
M1	Project Leader	Manager	8
M2	Domain Expert	Information Systems	14
M3	Evaluator	Technical	6
M4	Evaluator	Technical	5
M5	User	Non technical	9
M6	User	Non technical	6

These team members were then invited for a meeting to inform their tasks and responsibilities; make them understand the framework processes; and to set up and demonstrate the DM-PT. Table 6.23 are the main information related to this activity.

Table 6.23

Forming the Evaluation Team

Forming the Evaluation Team	
Aims	To identify of the roles and the evaluation team members.
Inputs	The project target and the stakeholders inside and outside the computer and IT center.
Outputs	The main roles in the project, the evaluation team members and their tasks.
Duration	1 day
Roles	Computer center director, competent unit head and expert
Used technique	JAD technique

C. Creating the Project WBS

In this third activity, a meeting was held to establish the WBS of the project. Among the tasks were to determine the project sources such as the required budget and time

to complete the project; and the project objectives. At the end of this meeting the project WBS was entered to the DM-PT as shown in Figure 6.19.

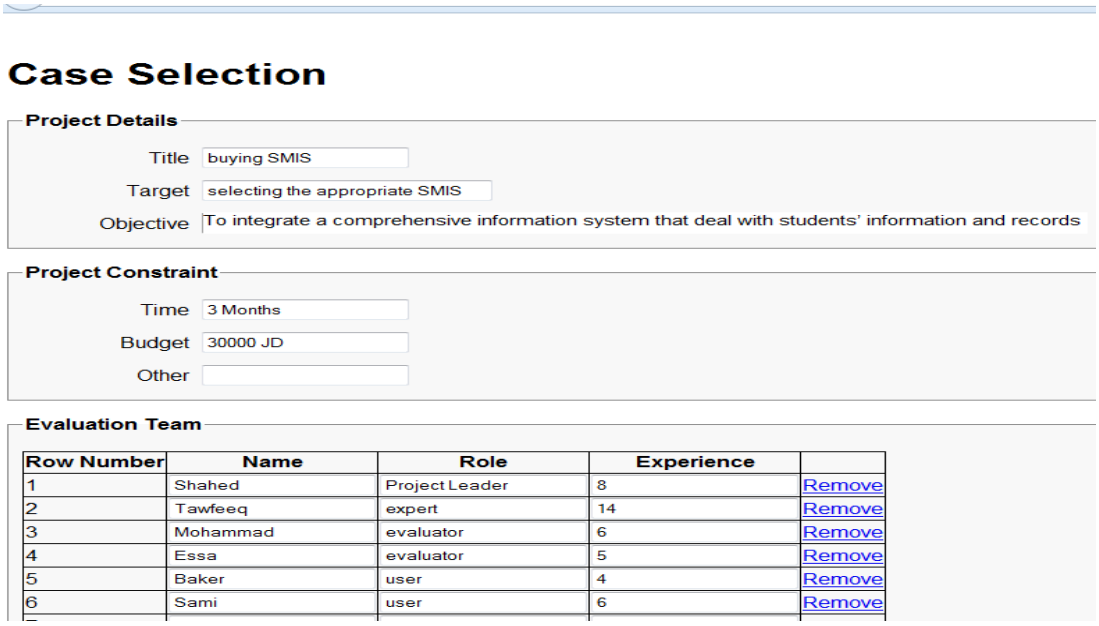


Figure 6.19. Snapshot Screen of the WBS Information

This activity is summarized in Table 6.24.

Table 6.24

WBS Creation

	WBS Creation
Aims	To determine the main information related to the project WBS such as the objectives and project constraint (budget and time).
Inputs	The project target, evaluation team.
Outputs	Project WBS (refer to Figure 6.19)
Duration	1 day
Roles	Project leader, evaluator, user and expert.
Used technique	JAD technique

6.3.1.3.3 Preparation Process

The initial data was established during the preparation process through three activities: defining the functional requirements, searching the COTS alternatives, and defining the yardstick.

A. Identifying Functional Requirements

The functional requirements in this project were identified during this activity. The first meeting was conducted with the users and stakeholders of the team to elicit the functional requirements. In the second meeting with other team members, the identified functional requirements were discussed and analyzed. These requirements were then entered into the DM-PT tool as shown in Figure 6.20. The full descriptions of the functional requirements are in Appendix I.

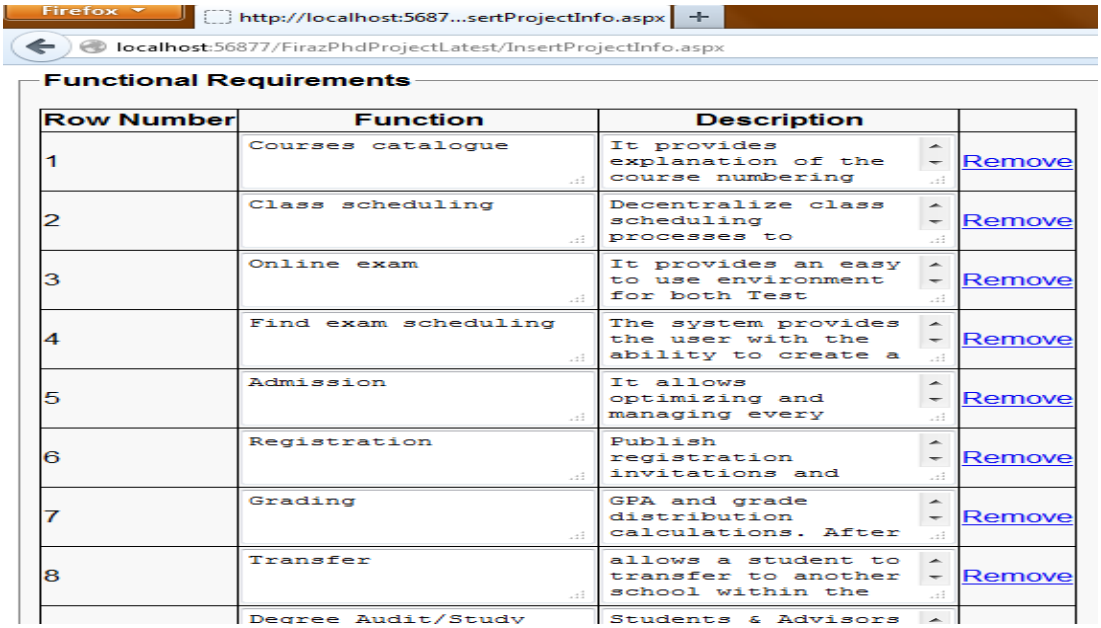


Figure 6.20. Snapshot Screen for the Functional Requirements

The set of key functional requirements identified in this activity was used as searching criteria in the next activity (searching activity). Table 6.25 presents the main information related to this activity.

Table 6.25

Defining the Functional Requirements Activity

Defining the functional requirements	
Aims	To identify the functional requirements of the target software.
Inputs	Project target, evaluation team, stakeholders, system documents
Outputs	Functional requirements (*refer to Figure 6.20*)
Duration	7 days
Roles	Evaluator , expert and user
Used technique	JAD technique and document review technique

B. The COTS Searching Activity

The potential SMIS alternatives were identified based on the functional requirements identified in the previous activity. Since the evaluators relied on the local market to find local vendors, the local inventory and market searching through a list of identified vendors in the market were used. From this search, four SMIS alternatives were identified. The main information of these application alternatives was collected from many sources such as from the software products' attached documents and websites. The SMIS alternatives and their initial information were then entered to the DM-PT as shown in Figure 6.21.

Figure 6.21. The SMIS Alternatives

The COTS searching activity is summarized in Table 6.26.

Table 6.26

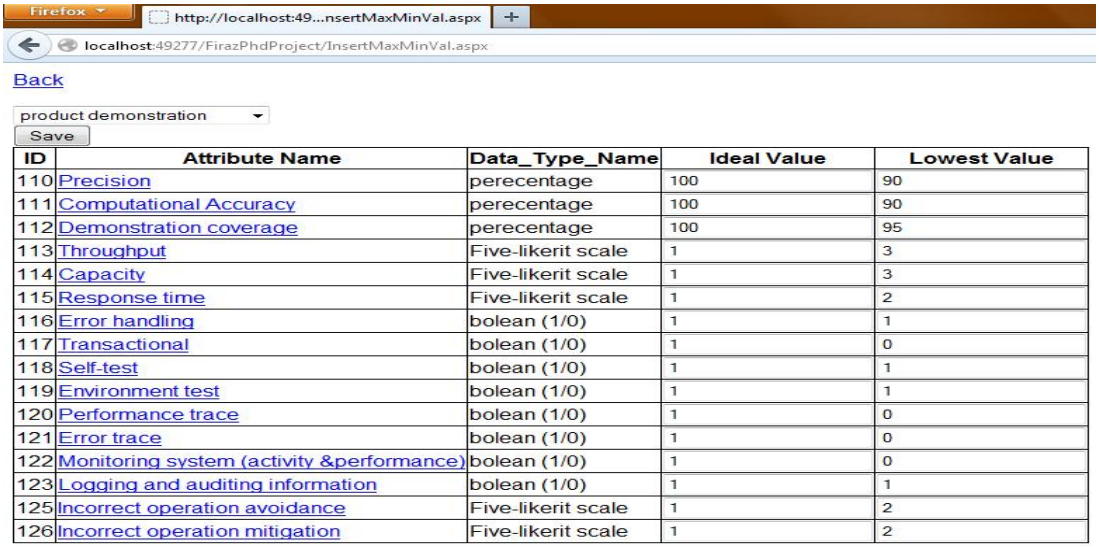
The COTS Searching Activity

	COTS Searching
Aims	To identify the potential SMIS software alternatives that meets the searching criteria (functional requirements).
Inputs	Searching criteria, searching sources (local inventory and the local market), and evaluation team.
Outputs	Alternatives list of SMIS software (refer to Figure 6.21).
Duration	6 days
Team members	Evaluator
Used technique	Local vendor advertising and shows and documents analysis.

C. Defining the Yardstick

The yardstick thresholds values were determined from a discussion among the domain expert, project leader, evaluator, and user. The values included the ideal and lowest values of the various attributes in the yardstick. The DM-PT was used during the discussion to facilitate the team in entering and saving the thresholds values of the yardstick in a systematic way. Besides the threshold values, the entire yardstick attributes, their explanations, and their data types were also stored. Figure 6.22

displays the snapshot screen of entering a group of the yardstick thresholds values in this project. The rest of the values were illustrated in Appendix I.



ID	Attribute Name	Data_Type_Name	Ideal Value	Lowest Value
110	Precision	perecentage	100	90
111	Computational Accuracy	perecentage	100	90
112	Demonstration coverage	perecentage	100	95
113	Throughput	Five-likerit scale	1	3
114	Capacity	Five-likerit scale	1	3
115	Response time	Five-likerit scale	1	2
116	Error handling	bolean (1/0)	1	1
117	Transactional	bolean (1/0)	1	0
118	Self-test	bolean (1/0)	1	1
119	Environment test	bolean (1/0)	1	1
120	Performance trace	bolean (1/0)	1	0
121	Error trace	bolean (1/0)	1	0
122	Monitoring system (activity &performance)	bolean (1/0)	1	0
123	Logging and auditing information	bolean (1/0)	1	1
125	Incorrect operation avoidance	Five-likerit scale	1	2
126	Incorrect operation mitigation	Five-likerit scale	1	2

Figure 6.22. The Yardstick Thresholds Values

Table 6.27 shows the main information required in the activity of defining the yardstick thresholds.

Table 6.27

Defining the Yardstick Thresholds Activity

	Defining the yardstick thresholds
Aims	To identify the thresholds values (ideal & lowest values) of the yardstick relating to the SMIS applications evaluation.
Inputs	The yardstick attributes and their data types (which provided by DM-PT) and evaluation team.
Outputs	Thresholds values for the attributes in the yardstick were determined (refer Appendix I).
Duration	2 days
Team members	Expert, project leader, evaluator, and user
Used technique	Local Inventories searching, and local market searching

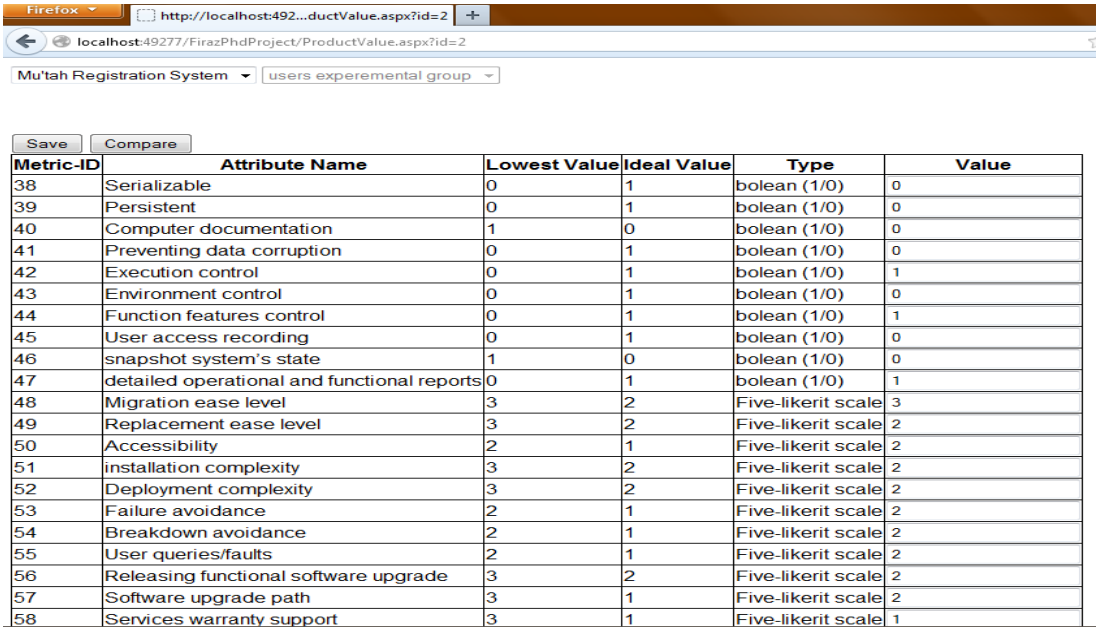
6.3.1.3.4 Evaluation and Selection Process

This evaluation and selection process aimed to collect and synthesize the data about the SMIS alternatives in order to select the fittest SMIS software. This process was carried out through the data collection and filtering, and decision making activities. The collected data was then short-listed into a few alternatives so that a decision can be made to select the fittest SMIS application. More description about these activities is as the following.

A. Data Collection and Filtering Activity

The first activity of the evaluation and selection process was the gathering of data related to the SMIS alternatives. Those alternatives that did not meet the yardstick thresholds values were filtered and removed from the list. In this activity, there are four levels of filtering depending on the data resources. The data at each level was then entered into the DM-PT. Finally, a short report was generated that shows the status of the software products either pass or fail as well as the reasons of failure.

In the first level, the SMIS alternatives related information was collected from the existing documents and the websites. Figure 6.23 displays the keyed-in data in the DM-PT.



Metric-ID	Attribute Name	Lowest Value	Ideal Value	Type	Value
38	Serializable	0	1	bolean (1/0)	0
39	Persistent	0	1	bolean (1/0)	0
40	Computer documentation	1	0	bolean (1/0)	0
41	Preventing data corruption	0	1	bolean (1/0)	0
42	Execution control	0	1	bolean (1/0)	1
43	Environment control	0	1	bolean (1/0)	0
44	Function features control	0	1	bolean (1/0)	1
45	User access recording	0	1	bolean (1/0)	0
46	snapshot system's state	1	0	bolean (1/0)	0
47	detailed operational and functional reports	0	1	bolean (1/0)	1
48	Migration ease level	3	2	Five-likerit scale	3
49	Replacement ease level	3	2	Five-likerit scale	2
50	Accessibility	2	1	Five-likerit scale	2
51	installation complexity	3	2	Five-likerit scale	2
52	Deployment complexity	3	2	Five-likerit scale	2
53	Failure avoidance	2	1	Five-likerit scale	2
54	Breakdown avoidance	2	1	Five-likerit scale	2
55	User queries/faults	2	1	Five-likerit scale	2
56	Releasing functional software upgrade	3	2	Five-likerit scale	2
57	Software upgrade path	3	1	Five-likerit scale	2
58	Services warranty support	3	1	Five-likerit scale	1

Figure 6.23. The Collected Data in the Level Two of Filtering Activity

Figure 6.24 shows a snapshot screen of the first level results of data collection and filtering process. At this level, all the SMIS alternatives met the thresholds values and passed to proceed to the next level.

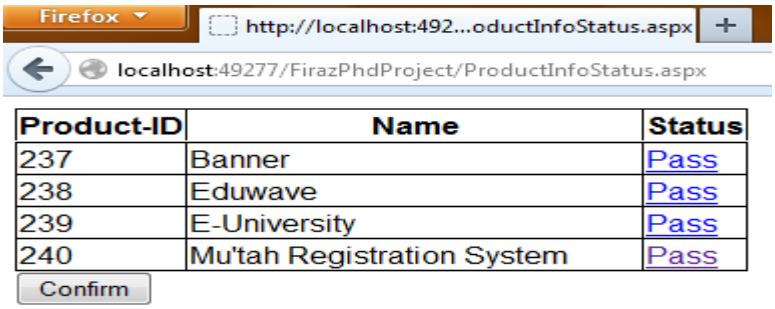


Figure 6.24. The Level One Filtering Result

In the second level, the evaluation form was used to collect data from the experimental users of the SMIS applications. The e-mail was used to contact the

experimental users and the evaluation forms were collected within 10 days. After entering the collected data into the DM-PT, the status of all the SMIS products were displayed as passed. In level three, the information related to the SMIS vendors was requested and collected from the vendors using the evaluation from. The output of this level indicated the status of one of the SMIS products (Banner software) as fail (Figure 6.25).

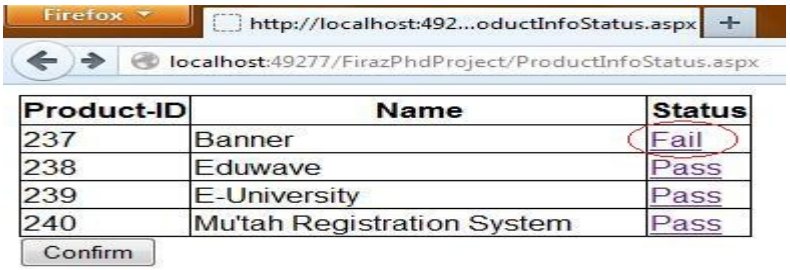
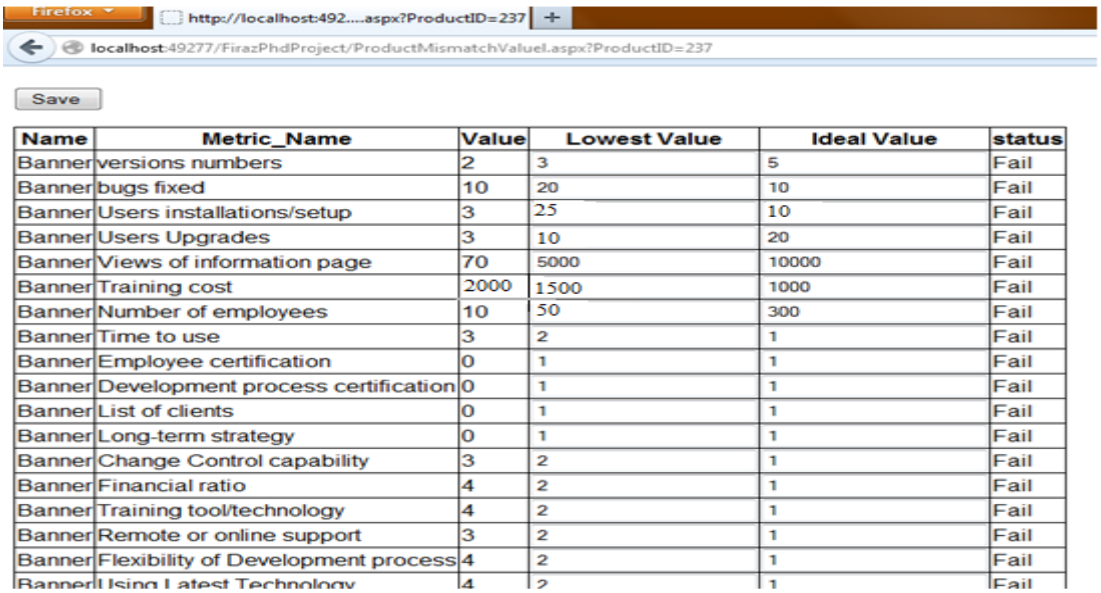


Figure 6.25. The Output of the Level Three

The various reasons of product failures are portrayed in Figure 6.26.



Name	Metric_Name	Value	Lowest Value	Ideal Value	status
Banner	versions numbers	2	3	5	Fail
Banner	bugs fixed	10	20	10	Fail
Banner	Users installations/setup	3	25	10	Fail
Banner	Users Upgrades	3	10	20	Fail
Banner	Views of information page	70	5000	10000	Fail
Banner	Training cost	2000	1500	1000	Fail
Banner	Number of employees	10	50	300	Fail
Banner	Time to use	3	2	1	Fail
Banner	Employee certification	0	1	1	Fail
Banner	Development process certification	0	1	1	Fail
Banner	List of clients	0	1	1	Fail
Banner	Long-term strategy	0	1	1	Fail
Banner	Change Control capability	3	2	1	Fail
Banner	Financial ratio	4	2	1	Fail
Banner	Training tool/technology	4	2	1	Fail
Banner	Remote or online support	3	2	1	Fail
Banner	Flexibility of Development process	4	2	1	Fail
Banner	Using Latest Technology	4	2	1	Fail

Figure 6.26. The Snapshot Screen of Product Failure

In the last level, the information related to the implementation was gathered by inviting the vendors, who still remain in the list, to demonstrate their software products in the presence of the evaluation team. The results of this level showed that only three SMIS products passed the filtering process (Figure 6.27).

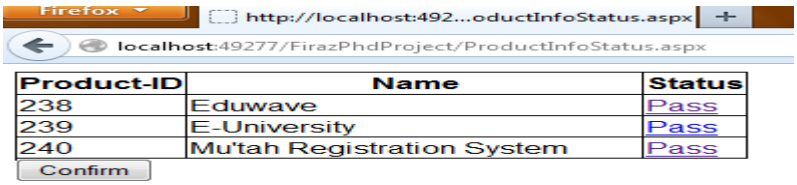


Figure 6.27. The Output of the Level Four

Since the products in Figure 6.27 fulfilled the yardstick thresholds values and met the functional requirements, further evaluation was needed to select the fittest SMIS product. This was performed in the next activity. Table 6.28 shows the main information related to the data collection and filtering activity.

Table 6.28

Data Collection and Filtering Activity

	Data Collection and Filtering Activity
Aims	To gather the data about the SMIS products and short-list them.
Inputs	The list of SMIS software products from the searching activity, yardstick thresholds values that provided by DM-PT, and evaluation team.
Outputs	Short List of SMIS software for further evaluation (refer to Figure 6.27).
Duration	28 days
Roles	Evaluator and user
Used technique	Evaluation form, software product demonstration attending, documents review and JAD technique to discuss the results.

B. Decision making Activity

This decision making activity was applied in this project for further evaluating the products list from the previous activity in order to select the most appropriate SMIS application to be used in organization B. The decision making technique applied in this activity involved two tasks: the CEC weighting and COTS alternatives scoring.

i) CEC Weighting

As mentioned earlier, the weights identification is very important to distinguish the evaluation criteria according to their importance. Therefore, the domain expert, project leader, user, and evaluator were required to meet and provide their judgments regarding the evaluation criteria through several pairwise comparisons at each level in the CEC. The DM-PT was keep running by the evaluation team during the meeting in order to directly key-in and check the consistency of these judgments. Figure 6.28 displays a snapshot screen of the evaluation criteria weightage that relate to the quality characteristics category in the second level of the CEC.

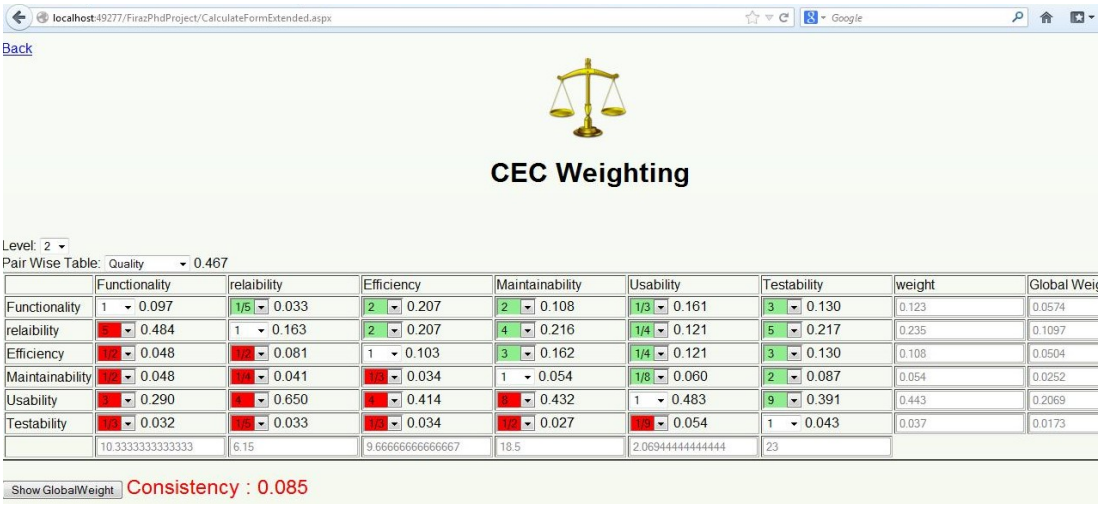


Figure 6.28. Criteria Weighting from the Second Level of the CEC

The weights of all criteria in the CEC are displayed in Appendix I. Table 6.29 presents the main information related to this task.

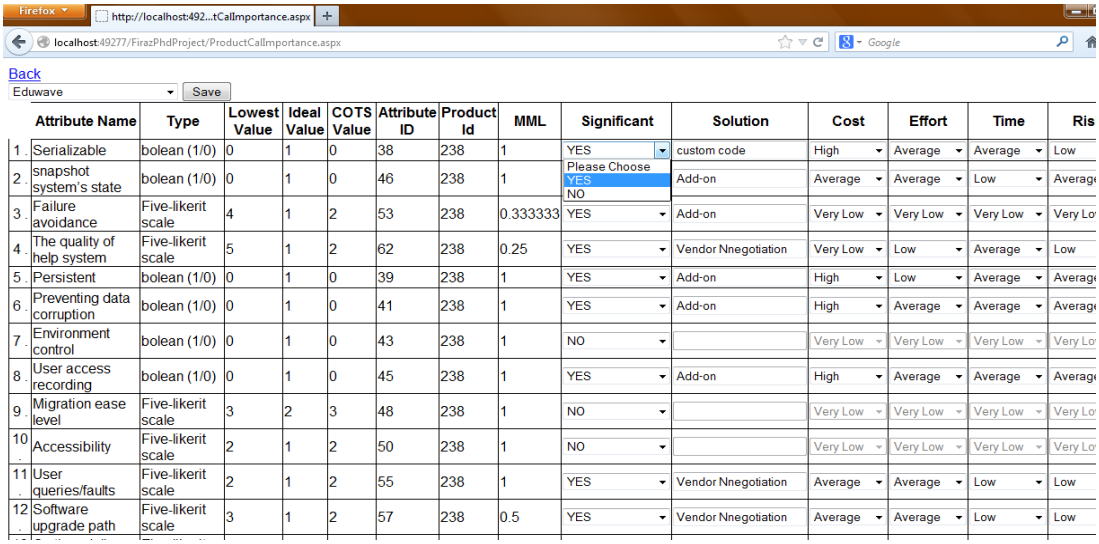
Table 6.29

Main Information Related to the CEC Weighting Task

CEC weighting task	
Aims	To assign weights to the criteria in the CEC.
Inputs	The CEC criteria (which provided by DM-PT) and evaluation team.
Outputs	CEC weights (Appendix I).
Duration	1 day
Team members	Expert, project leader, evaluator, and user
Used technique	JAD technique to conduct the meetings, DM-PT

ii) **COTS Alternatives Scoring**

The COTS alternatives scoring task aimed to calculate the final scores for each SMIS alternative against the CEC. The calculation was performed by handling the SMIS software products mismatches, potential solution and their constraints (cost, effort, time, and risk). This was based according to the decision making technique in the framework. In this context, the DM-PT was used to carry out all the required calculations. In this technique, the role of the evaluation team was just limited on determining the significant detected mismatches for each SMIS software product. Following this, the team was also required to identify the mismatches potential solutions as well as their respective constraints. Figure 6.29 displays a snapshot of the SMIS software (Eduwave software) mismatches together with their solutions and constraints.



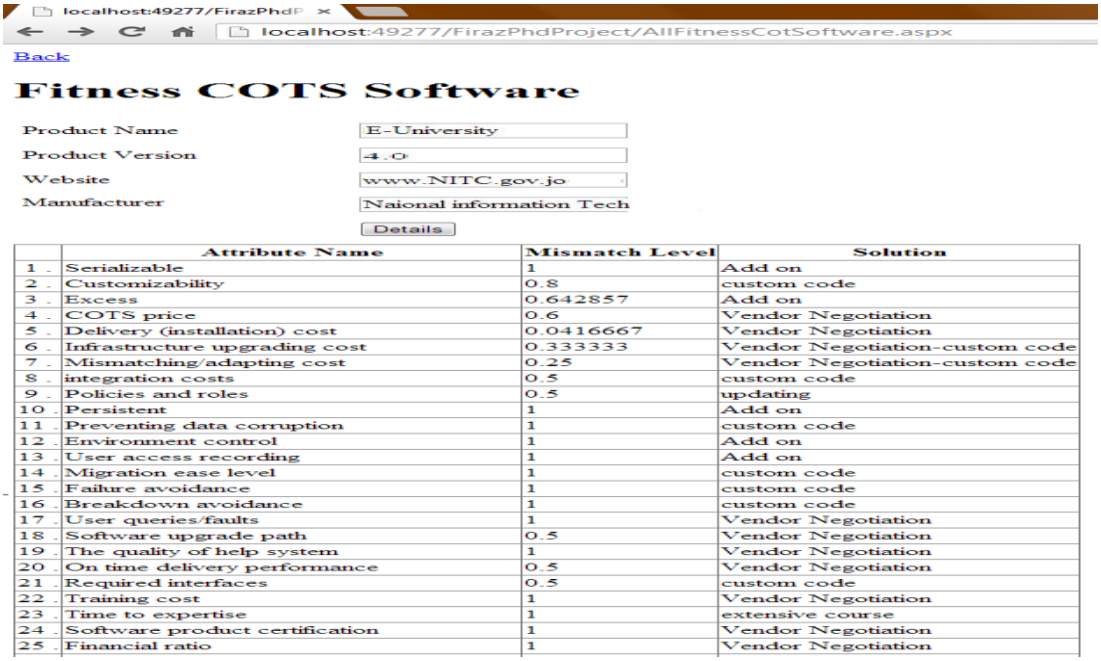
Back

Eduwave Save

	Attribute Name	Type	Lowest Value	Ideal Value	COTS Value	Attribute ID	Product Id	MML	Significant	Solution	Cost	Effort	Time	Ris
1	Serializable	bolean (1/0)	0	1	0	38	238	1	YES	custom code	High	Average	Average	Low
2	snapshot system's state	bolean (1/0)	0	1	0	46	238	1	Please Choose / YES / NO	Add-on	Average	Average	Low	Averag
3	Failure avoidance	Five-likerit scale	4	1	2	53	238	0.333333	YES	Add-on	Very Low	Very Low	Very Low	Very Lo
4	The quality of help system	Five-likerit scale	5	1	2	62	238	0.25	YES	Vendor Nnegotiation	Very Low	Low	Average	Low
5	Persistent	bolean (1/0)	0	1	0	39	238	1	YES	Add-on	High	Low	Average	Averag
6	Preventing data corruption	bolean (1/0)	0	1	0	41	238	1	YES	Add-on	High	Average	Average	Averag
7	Environment control	bolean (1/0)	0	1	0	43	238	1	NO		Very Low	Very Low	Very Low	Very Lo
8	User access recording	bolean (1/0)	0	1	0	45	238	1	YES	Add-on	High	Average	Average	Averag
9	Migration ease level	Five-likerit scale	3	2	3	48	238	1	NO		Very Low	Very Low	Very Low	Very Lo
10	Accessibility	Five-likerit scale	2	1	2	50	238	1	NO		Very Low	Very Low	Very Low	Very Lo
11	User queries/faults	Five-likerit scale	2	1	2	55	238	1	YES	Vendor Nnegotiation	Average	Average	Low	Low
12	Software upgrade path	Five-likerit scale	3	1	2	57	238	0.5	YES	Vendor Nnegotiation	Average	Average	Low	Low

Figure 6.29. The Eduwave Software Mismatches and Their Solutions

After determining the mismatches information for each SMIS in the list, the DM-PT would then compute the FFS and automatically select the fittest one. This is shown in Figure 6.30.



Back

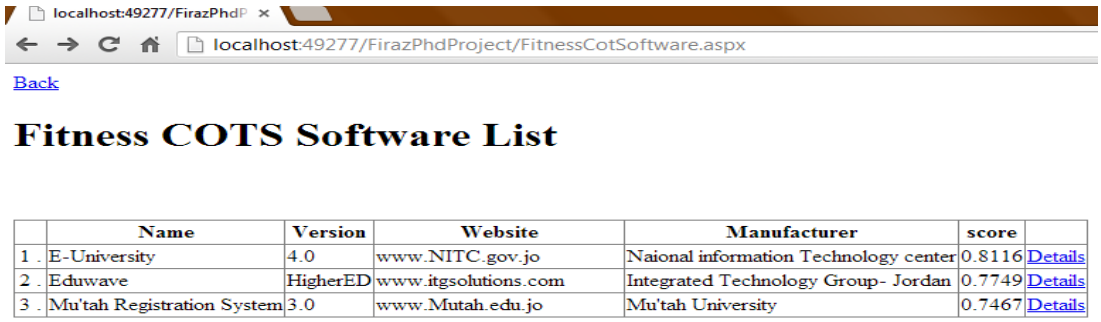
Fitness COTS Software

Product Name: E-University
Product Version: 4.0
Website: www.NITC.gov.jo
Manufacturer: Naional information Tech
Details

	Attribute Name	Mismatch Level	Solution
1	Serializable	1	Add on
2	Customizability	0.8	custom code
3	Excess	0.642857	Add on
4	COTS price	0.6	Vendor Negotiation
5	Delivery (installation) cost	0.0416667	Vendor Negotiation
6	Infrastructure upgrading cost	0.333333	Vendor Negotiation-custom code
7	Mismatching/adapting cost	0.25	Vendor Negotiation-custom code
8	integration costs	0.5	custom code
9	Policies and roles	0.5	updating
10	Persistent	1	Add on
11	Preventing data corruption	1	custom code
12	Environment control	1	Add on
13	User access recording	1	Add on
14	Migration ease level	1	custom code
15	Failure avoidance	1	custom code
16	Breakdown avoidance	1	custom code
17	User queries/faults	1	Vendor Negotiation
18	Software upgrade path	0.5	Vendor Negotiation
19	The quality of help system	1	Vendor Negotiation
20	On time delivery performance	0.5	Vendor Negotiation
21	Required interfaces	0.5	custom code
22	Training cost	1	Vendor Negotiation
23	Time to expertise	1	extensive course
24	Software product certification	1	Vendor Negotiation
25	Financial ratio	1	Vendor Negotiation

Figure 6.30. The Snapshot Screen for the Fittest SMIS Product

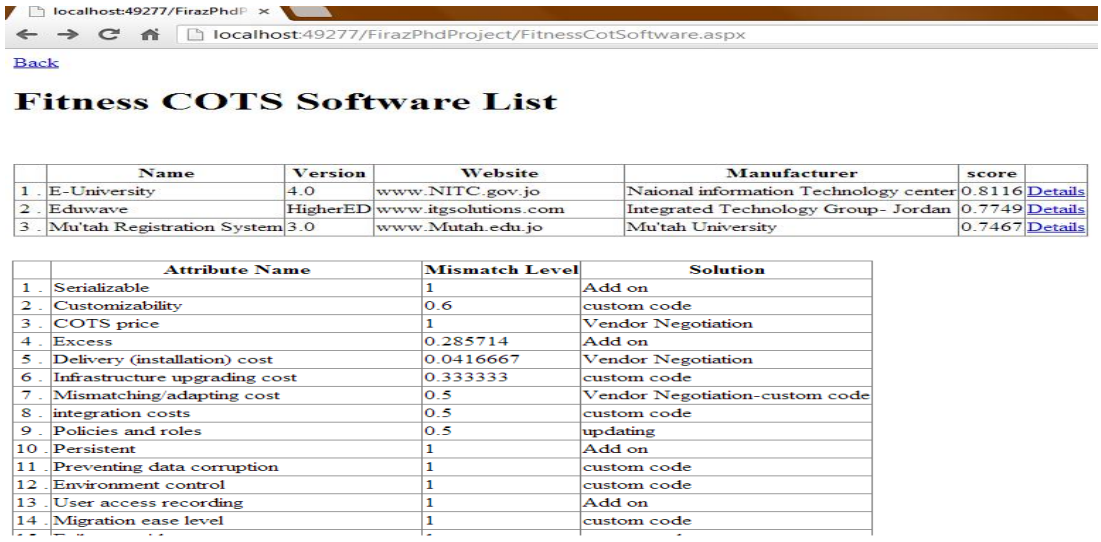
The DM-PT also presented a sorted list of the software products according to their fitness to the evaluation criteria (Figure 6.31).



	Name	Version	Website	Manufacturer	score	
1 .	E-University	4.0	www.NITC.gov.jo	Naional information Technology center	0.8116	Details
2 .	Eduwave	HigherED	www.itgsolutions.com	Integrated Technology Group- Jordan	0.7749	Details
3 .	Mu'tah Registration System	3.0	www.Mutah.edu.jo	Mu'tah University	0.7467	Details

Figure 6.31. Snapshoot Screen from the Sorted List of Products Screen

Figure 6.32 provides the report of the product mismatches together with their potential solutions.



	Name	Version	Website	Manufacturer	score	
1 .	E-University	4.0	www.NITC.gov.jo	Naional information Technology center	0.8116	Details
2 .	Eduwave	HigherED	www.itgsolutions.com	Integrated Technology Group- Jordan	0.7749	Details
3 .	Mu'tah Registration System	3.0	www.Mutah.edu.jo	Mu'tah University	0.7467	Details

	Attribute Name	Mismatch Level	Solution
1 .	Serializable	1	Add on
2 .	Customizability	0.6	custom code
3 .	COTS price	1	Vendor Negotiation
4 .	Excess	0.285714	Add on
5 .	Delivery (installation) cost	0.0416667	Vendor Negotiation
6 .	Infrastructure upgrading cost	0.333333	custom code
7 .	Mismatching/adapting cost	0.5	Vendor Negotiation-custom code
8 .	integration costs	0.5	custom code
9 .	Policies and roles	0.5	updating
10 .	Persistent	1	Add on
11 .	Preventing data corruption	1	custom code
12 .	Environment control	1	custom code
13 .	User access recording	1	Add on
14 .	Migration ease level	1	custom code

Figure 6.32. Snapshoot Screen from the Report Screen for Product 3

The findings from this activity were used in the integration phase where the selected product is integrated to the system in the organization. In addition, these reports were sent to the vendors as feedback in order to for them to improve their product in the

next versions. All reports for each product in the list are shown in Appendix I. The summary of this activity is presented in Table 6.30.

Table 6.30

Decision Making Activity

Decision Making Activity	
Aims	To select the fittest SMIS products and generate reports about the main results (e.g. mismatches, solution) for adaptation and integration phases.
Inputs	The short-listed software products from the data collection and filtering activity, the detected mismatches that provided by DM-PT, and evaluation team.
Outputs	The fittest SMIS software (refer to Figure 6.30), list of sorted software products (refer to Figure 6.31) and the reports for each software products (refer to Figure 6.32 and Appendix I).
Duration	1 day
Roles	Project leader, Evaluator and User
Used technique	Decision making and JAD techniques to review the results.

At the end of this activity, the evaluation team met again to review the results and prepare the final report for the Computer and Information Technology Center manager. The results were obtained by the project leader and the head of the in-charge unit in the Computer Center.

6.3.1.4 Discussion of the Findings

This section aims to present the results of evaluating COTS-ESF in term of its applicability in the above two case studies. The evaluation was conducted through an interview with the evaluation team in the two organizations based on a set of

evaluation criteria that have been identified from the literature, as explained in chapter 3. These criteria are gain, task support, and interface satisfactions.

1. Gain satisfaction

Gain satisfaction was used to measure whether COTS-ESF would be beneficial in the real life through the following criteria:

- Perceived Usefulness: The evaluation team agreed that the COTS-ESF was found useful for selecting the appropriate COTS software. The evaluation team from organization 'A' indicated that the variety of the evaluation criteria that provided by the framework to evaluate the COTS software increases the selection performance. The team in organization 'B' added that the structure of the evaluation team that comprised of different evaluation perspectives (e.g. domain expert, technical, user, and manager) from different stakeholders positively influenced the effectiveness of the members' roles. This enables the team to provide, analyze, and make accurate judgments during the evaluation and selection process. Furthermore, COTS-ESF was found to be very effective in terms of the time required to achieve the evaluation target and the efforts involved in choosing the appropriate persons to carry out such process.
- Decision support satisfaction: In this regard, the evaluation team from the two organizations stressed that the COTS-ESF achieved the decision support satisfaction by providing a well-defined decision making process structure conducted through the use of software tool. Furthermore, COTS-ESF has the potential to reduce the individual bias by using the JAD session that bring

together the expert, technician, user, and manager as a team. Such session provides them a chance to share their views, understand others views, and release the appropriate judgments during the decision making process.

- Comparison with the current selection method: The COTS-ESF is more suitable and accurate for evaluating and selecting the COTS software compared to the traditional method that is currently used in both organizations. Both evaluation teams admitted that the processes and activities of COTS-ESF are clear, consistent, and easy to apply, while the traditional methods are used without any well-defined processes and techniques. In addition, the selection processes and activities are supported by software tool which make it more systematic and reliable. Moreover, the evaluation criteria in this framework provide a more accurate technique in selecting the appropriate COTS software compared to the traditional methods that rely only on specific criteria (i.e. financial issues) and ignore other important criteria. In term of time, COTS-ESF managed to shorten the project time where most of it processes required around 17 days to complete, however, most of time was spent in collecting the data (more than one moth).

- Cost-effectiveness: COTS-ESF was found to be cost effective due to various reasons. For example, COTS-ESF selected the most appropriate COTS software that has few mismatches against the user requirements, which save the costs of COTS customization and integration in the next phases. Furthermore, by using several resources when searching the COTS alternatives, the organization 'A' team found that the COTS-ESF gives them a lot of selection choices. Having such choices is very cost effective because they can avoid buying the COTS product

from specific vendors that may go out of business or may not provide the after-sales support.

- Clarity (clear and illuminate the process): The framework processes were found to be very clear and understandable to the evaluation teams, where each process clearly presents the required inputs, outputs, methods or techniques, and roles. This prevents any overlapping roles.
- Task Appropriateness: COTS-ESF is appropriate for evaluating and selecting the COTS software in a very systematic and effective way compared to the ad-hoc manner which had been used before. In addition, the CEC in this framework were found to be more suitable and comprehensive in evaluating and selecting the appropriate COTS software according to the user requirements.

2. **Interface satisfaction**

Interface satisfaction criteria are used to measure the interface characteristics in term of presentation, format, and processing efficiency. These criteria are discussed as follows:

- Perceived ease of use: According to the responses of the evaluation teams, COTS-ESF was perceived as easy to use because it uses well-defined processes, activities, and techniques. Moreover, it is conducted using the software tool (DM-PT) which helps the evaluation team to manage, input, and filter the data easily and efficiently. In fact the calculations, comparisons, and sorting can be performed in a systematic manner.

- Internally consistent: COTS-ESF was internally consistent because the components of the framework complement each other. Precisely, it starts with the planning process where the evaluation target is defined, the evaluation team is forming, and then the WBS is established. These activities are to control all the processes in COTS-ESF. After that, the pre-evaluation information is prepared in the preparation process stage where the functional requirements are defined and the COTS alternatives are identified. Then, the process of evaluating and selecting the fittest COTS software is carried out. In addition, the DM-PT helps in maintaining the consistency through the entire process.

- Organization (well organized): The framework was found to be well organized and structured where the flow of the information and the sequence of the framework processes, activities and task, and the evaluation team roles were sorted and organized in a clear and understandable manner. This will surely ease the evaluation and selection process even though the project is complicated.

- Appropriate for audience: The evaluation teams in both organizations indicated that COTS-ESF was appropriate for the audience. Those audience are referred to the team of evaluators that often have different skills (technical and non technical). The appropriateness is realized because the framework provides variety of roles (e.g. judge, search, collect the data, and manage) depending on the evaluators' skills. Moreover, the used of the DM-PT as the support tool satisfies the members of the evaluation teams.

- Presentation (readable and useful format): All respondents indicated that the COTS-ESF produced the results in a readable and useful format. The DM-PT manages to present a very apparent and understandable the results.

3. Task Support Satisfaction

The task support satisfaction measures whether COTS-ESF attains its anticipated objectives and satisfies its evaluators. The measurement includes the following criteria:

- Ability to produce expected results: COTS-ESF is able to produce expected results. This was stressed by both the evaluation teams whereby COTS-ESF proved to reflect high capabilities in producing accurate results within a short time compared to the previous methods. The accurate, reliable, and satisfied results are able to be delivered because COTS-ESF provides a well-defined sequence of activities and tasks, a wide variety of evaluation criteria, several COTS software searching resources, ways of handling the COTS mismatches, and DM-PT to perform the computations.
- Ability to produce usable results: The respondents indicated that COTS-ESF is able to produce usable results. The selected COTS software products by both teams were accepted by their management to be used in their organizations. The selection was recognized because the evaluation teams members had be chosen based on several skills and experiences. To have such skillful teams made the management more confidence with the selection

decision compare to the previous methods which involved only one individual.

- Completeness (adequate or sufficient): COTS-ESF was found to be adequate and sufficient in evaluating and selecting the COTS software in these organizations. The respondents indicated that COTS-ESF provides a set of evaluation criteria that are sufficient for such evaluation. Among the criteria were the product, user organization and vendor characteristics; the framework processes; decision making technique; and the DM-PT support tool.
- Ease of implementation: The evaluation teams agreed that COTS-ESF was easy to implement. The respondents explained that the inputs, outputs, techniques, and the roles of the team members led to an easier and more applicable evaluation and selection process. Moreover, the DM-PT facilitates the implementation of COTS-ESF by delivering reliable results efficiently in terms of the time and efforts.
- Understandability (simple to understand): COTS-ESF was found to be readable and understandable. The evaluation teams asserted that the processes in the framework were organized and labeled in such a way that made them simple to understand. The decision making technique and CEC presented by the DM-PT is easy to understand and apply.

Based on the above responses, COTS-ESF is effective and applicable in the real life.

6.3.2 Yardstick Validation

Since the objective of the validation process is to demonstrate the correctness for its intended purpose, the comparison with existing ideal work or baseline model is also considered as the reliable and perfect way to validate a model which it is called the yardstick validation (carson, 2002). Therefore, the yardstick approach is used usually with others validation approaches to increase the trustworthiness of the model validation process (Sargent, 2011). In particular, if the model's components are compared and found that they coincide with baseline models in the same field, it will be considered as evidence to the validity and correctly of the model (carson, 2002; Sargent, 2011).

The yardstick validation was conducted through the following steps:

Step1: Identify the baseline models for COTS software evaluation and selection. As discussed in Chapter 2, there are three main models that commonly and usefully used or referred as the baseline models in COTS software evaluation and selection (George et al., 2008; Javed et al., 2012). These models are: OTSO, PORE, and STACE.

Step2: Determine the comparison criteria. COTS-ESF was compared with the baseline models based on their strengths and weaknesses. Since the evaluation theory provides the basic six evaluation components for any evaluation process, these strengths and weaknesses of the models were investigated according to those components, which are: determining the evaluation target, the proposed evaluation

criteria, yardstick, the proposed data collection techniques, the used synthesis technique, and the proposed evaluation process.

Step3: Conduct the comparison. As mention previously, the strengths and weaknesses for each model were determined based on investigating the six basic evaluation components. Precisely, the strengths are identified when the model address one of the basic evaluation components, while the weakness describes the lack of addressing them. Table 6.31 presents the comparison between COTS-ESF and the baseline models based on their strengths and weaknesses.

Concerning the delimitating of the evaluation target, each of the PORE, STACE, and COTS-ESF focus on the COTS software as the target evaluation. The OTSO model handles the OTS components that involved two different kinds of components with different characteristics, which are: COTS and OSS components. Therefore, addressing different kinds of software will reflect negatively on the accuracy of the evaluation and decision making processes.

The evaluation criteria have been proposed by all of the models. However, the OTSO and PORE models focused on the functionality and quality criteria while they ignored other important criteria such as the domain and organization criteria. The STACE model stressed on the social and technical criteria and disregarded the domain criteria that play important role in COTS software evaluation process. As for COTS-ESF, the non-functional requirements were addressed well by proposing CEC that proposes the required criteria for COTS software evaluation.

Despite the yardstick or the standard component is proposed as one of the basic evaluation components for any evaluation process, it is not considered by the previous models (OTSO, PORE, and STACE) when evaluating and selecting COTS software. Defining the yardstick helps to identify the ideal state of user's requirements which, in turn, helps to filtrate the COTS software alternatives, identify the mismatches between this state and the COTS alternatives, and finally select the fitness COTS alternative. Therefore, COTS-ESF takes the advantages of the yardstick component by adapting it within the evaluation process to achieve accurate results.

With respect to the data collection techniques, the explicitly defining these techniques makes the evaluation process more understandable and easy to implement. In this regards, the OTSO does not explain how to collect the evaluation data and what the used techniques, while the PORE and STACE models suggested variety of general techniques for data collection without showing when and how to use them during the evaluation process. In contrast, COTS-ESF provides a clearly description about the data collection techniques. These techniques, such as JAD and evaluation form, have been identified based on three sources of data which are: related documents, experimental users group, vendor, and COTS demonstration to collect all required evaluation data related to the attributes in CEC in order to provide the accurate and reliable evaluation process.

In any evaluation process, the decision making technique is the most critical component where the final selection decision is made. Each of the OTSO, PORE,

and STACE models relies on the MCDM such as AHP and WSM techniques to analysis the data and makes the decision. However, these techniques have several limitations such as difficult to apply when the number of criteria and COTS alternatives are increased, and have deficiency in dealing with COTS mismatches (refer to the Table 2.3). COTS-ESF handled the limitations in the previous techniques and proposed new decision making technique based on integrating AHP and GA techniques to address the COTS mismatches and provide accurate decision making process. This technique was also supported by software tool (DM-PT) that is missing in the previous models to facilitate its implementation.

The evaluation processes play important role to manage and connect between the other components of evaluation. In this regards, the OTSO model provide a well-defined evaluation processes and tasks to address the complexity of the evaluation process, and provide a model for cost-benefits estimation. The PORE is called the requirements engineering approach since it emphasizes on the requirements engineering, while the iterative process that proposed by this model become more complex when the number of COTS alternatives are increasing. The STACE model also provide set of evaluation processes such as the requirements elicitation and defining social-technical criteria, with stressing on inclusion social-technical issues and users participation. This makes it more costly and time consuming especially when the participating users do not play effective role in the evaluation process. Comparing with previous models, the evaluation processes that have recommended by COTS-ESF are more inclusive, where the lack of explicitly addressing some of the processes and tasks in the previous models has been handled. For example, the

planning process that is not clearly addressed by the previous models was defined through several tasks such as delimitating the evaluation target, and forming the evaluation team. In addition, the data collection and filtering task is also not sufficiently handled in the previous models, while COTS-ESF provides further details about its required steps, levels, and used techniques.

In conclusion, COTS-ESF was built based on the previous baseline models where the strengths of these models were taken into account and the weaknesses were handled in order to empower it. For instance, COTS-ESF provides a systematic and well-defined evaluation. its selection process based on the evaluation theory. In addition, the framework addressed the non-functional requirements through proposing the CEC to tackle all the required COTS evaluation criteria. To provide the accurate decision making, COTS-ESF proposed new technique that was developed based on the integrating of AHP and GA techniques to address COTS mismatch issue and selecting the fitness COTS software. Finally, the DM-PT tool was developed to support the framework implementation in systematic and efficient way.

However, as with all theoretical models, COTS-ESF has some limitations including that of required more time for that data collection and it seems more suitable for large project that small one. Notwithstanding the limitations of COTS-ESF, it has more strengths points than previous models and thus it was found valid and useful to evaluate and select COTS software.

Table 6.31

Comparing COTS-ESF with Other Baseline Models for COTS Software Evaluation and Selection

The Model	Strengths	Weaknesses
COTS-ESF	• It starts by delimitating the evaluation target that represents by the COTS software. • It provides the required evaluation criteria that is called CEC based on addressing non-functional requirements • It provides the yardstick or standard that represents the ideal case for conducting the evaluation which supports to select the fitness COTS product. • It also determines a set of data collection techniques (e.g. evaluation form, JAD, and COTS demonstration) based on different sources of data related to the evaluation criteria which make the evaluation process more reliable. • It provides accurate decision making techniques by taking advantages of AHP and GA techniques to address COTS mismatches (identify the mismatches, the resolution actions, and their constraints) and provide accurate decision. • It supports systematic, accurate, and well-define evaluation process through providing three main processes (planning, preparation, and evaluation & selection processes) with their activities and tasks supported by software tool which make it easy to implement.	• The CEC model uses many attributes in low level which make the data collection is time consuming task. • It is more suitable for large projects. • Defining the yardstick values by the evaluation team does not often reflect the values of the COTS software features in the market. • It focuses on selecting single COTS software for each project.
OTSO	• It addresses the complexity of COTS software evaluation by showing more explanation about the processes and tasks of the software selection such as including entry and exit criteria. • Provides hierarchical and detailed definition of set of evaluation criteria that are derived from reuse goals. • Provides a model for costs-benefits estimation for each alternative. • Relies on the decision-making techniques (WSM & AHP) to analyse and summarise evaluation results.	• It is developed to evaluate the OTS component that consists of two different kinds of components with different aspects, which are COTS and OSS. • Lack of addressing non-functional requirements such as organizational and domain requirements. • It does not provide a standard or yardstick for conducting the evaluation. • The data collection techniques are not clearly specified. • AHP and WSM techniques are proposed as decision making techniques suffer from many limitations such as they do not address the COTS mismatches and only appropriate for few comparisons (refer to Table 2.3). • The planning process is not adequately addressed such as the evaluation team formation, project constraints, and evaluation target are not explicitly defined.
PORE	• The COTS software is used as the evaluation target of this approach. • It provides guidance to model requirements for COTS software selection. • It provides a variety of methods and techniques for requirements engineering, data	• Proposing some of non-functional criteria such quality criteria and ignore other important criteria such as the domain and vendor. • Lack of using the yardstick during the evaluation process.

	collection, and data analysis such as feature analysis and MCDM techniques. • It is considered as a template-based method where the used templates provide guidelines for conducting the evaluation process. • It provides a parallel and an iterative requirements acquisition and product selection/rejection process.	• Several techniques of data collections were proposed without specifying when and how to use them. • Use of traditional approaches and techniques that inadequately address COTS mismatches for decision making that makes it vulnerable to make inaccurated selection decision. • It is considered as the requirement engineering approach, since it is more focused on acquiring the requirements. • The iterative process that is proposed by this method will be more complex and inefficient process when there is a increasing number of alternatives and requirements which makes it labour intensive.
STACE	• It supports the evaluation of both COTS software and the underlying technology as the evaluation target. • It uses social-technical techniques (i.e., social-technical criteria and user participation) to improve the COTS software selection process. • It provides a set of well-defined evaluation processes such as the requirements elicitation, social-technical criteria definition, and evaluation process.	• Despite of many social-technical criteria were proposed to handle non-functional requirements, there are an important criteria that is not addressed such as the domain criteria. • It ignored the important role of using yardstick to evaluate and select the fitness COTS software. • Relies on AHP for decision making where some aspects of AHP is having subjective bias and does not handle the COTS mismatches. • It increases the cost of the evaluation process because of inclusion of nontechnical issues and user participation that may not make an effective contribution in COTS selection. • It does not adequately deal with evaluation of software for smaller organisations or projects.

6.4 Summary

This chapter discusses COTS-ESF evaluation process that was presented through the verification and validation stages. In the verification stage, COTS-ESF has been verified by 12 experts. The Delphi technique was used to conduct the verification stage through three rounds of modifications in order to improve the framework. In the final round, the framework components were modified and accepted according to the improvements agreed by the experts.

In the validation stage, two approaches were used in this stage: case study approach and yardstick validation approach. In regards with case study approach, two case studies from Malaysia and Jordan were used to apply COTS-ESF in order to examine its applicability across two different countries. To do so, the DM-PT was developed to facilitate COTS-ESF implementation using the ASP.Net and VB.Net technology as the programming language. It was found that COTS-ESF was applicable in the real world. This was based on several evaluation criteria to evaluate its real life applicability, which are: gain, interface, and task support satisfactions. In the yardstick validation, COTS-ESF was compared with OTSO, PORE, and STACE as a common and baseline models in the COTS software evaluation and selection process. The comparison has been conducted based on investigating the strengths and weaknesses of those models. The result of the evaluation process is COTS-ESF which was found valid and applicable for evaluating and selecting the COTS software in the real world.

CHAPTER SEVEN
THE CONCLUSION AND FUTURE WORK

7.1 Introduction

This chapter presents the conclusion of the overall work during this research by presenting the main findings of the studies that were used through developing COTS-ESF. This chapter also discusses the research contributions and presents the potential directions of the future work related to the field of COTS software evaluation and selection. This chapter ends by presenting the final conclusion.

7.2 General Discussion

As the main objective, this research aims to develop a new framework for the COTS software evaluation and selection process. The overall research process for developing and estimating the new framework was carried out through a theoretical study, empirical study, framework development, and framework evaluation. These studies are described in the following sections.

7.2.1 Theoretical Study

The literature review represents the starting point for providing knowledge and background of any research problem. Most of the academic and practiced literature related to CBS, COTS software evaluation and selection, evaluation criteria, and other relevant topics were reviewed and analyzed. Consequently, the theoretical knowledge was established in the field of CBD and the research gap was identified in the COTS software evaluation and selection. Typically, the literature on the COTS software evaluation and selection was addressed deeply where its main problems and challenges were highlighted. Most of these problems are related to the lacking of the

following: adequate COTS software evaluation criteria; well-defined, systematic, and repeatable COTS software evaluation and selection process; and COTS mismatches management. The existing COTS selection methods such as the OTSO and PORE were studied and criticized in order to identify common limitations and problems. Besides the common processes and activities, the discussion also touched on the MCDM method by emphasizing on the used of AHP technique in supporting the COTS selection. In addition, the mismatches concepts, types, and techniques (e.g. GA technique) were deeply discussed in order to handle the COTS mismatches problem. In addressing the non-functional requirements problem, several models and methods, such as Q'facto 12 and Bertoa's quality models, were studied, analyzed, compared, and synthesized to identify the common evaluation criteria related to the COTS software.

The main findings of this study have identified number of problems; the main processes, activities and techniques; a set of the evaluation criteria; existing methods and models associated with COTS selection, and the advantages and disadvantages of using the COTS software.

7.2.2 Empirical Study

This study aims to elicit and synthesize the current practices related to the COTS software evaluation and selection in the Jordanian organizations. In addition, it is also conducted to investigate the importance of the processes and activities, techniques, methods, evaluation criteria, benefits and risks of CBS as identified in the previous theoretical study. In carrying out this empirical study, the survey approach was adopted and the questionnaire was used to collect the data.

The main findings of this survey indicate that the main challenges and problems faced in the real world include the lack of using formal methods in evaluating and selecting the COTS software, COTS mismatches, and lack of handling the non-functional requirements. Instead of using any existing methods, most organizations prefer to rely on ad-hoc manner in selecting the COTS software. Generally, the ad-hoc manner is conducted based on the experiences of individual managers and developers, or the relationships with particular vendors.

The survey results also point out the main activities that were carried out by organization during the COTS software selection process. The activities are planning, searching, screening, and defining the evaluation criteria. There are a number of techniques being used by the organizations in performing the evaluation and selection process. For example, in defining the evaluation criteria, most organizations choose the documents review technique. The same technique is also applied for collecting data as well as the experimental users group and by attending the COTS demonstration. For the data analysis purposes, most organizations in the survey depend on their user experience and attending the software demonstration.

The survey also stresses on the importance of taking into account the non-functional requirements in evaluating and selecting the COTS software. Five categories of such requirements that have been identified are: i) quality, ii) domain, iii) architectural, iv) operational environment, and v) vendor. The requirements under the quality category are functionality, efficiency, reliability, maintainability, usability, and testability, while for the domain includes security, popularity and maturity. The architectural category looks into reusability, portability, interoperability, safety in use and operation support. The operational environment category consists of the system

platform characteristics and existing software development environment, culture and financial issue. Finally, the vendor category is about vendors' reputation, stability and supportability. Other findings as revealed from the survey are related to the CBS development approach, benefits, and risks. The "purchase and adapted" is the common approach for building the CBS, while its main benefit and risk are the reduction of the development cost and the lack of support when the vendor goes out of business.

The major outcomes of this study are the identification and classification of the core activities, techniques, and criteria that support the COTS software evaluation and selection process. This study brings out a documented and better understanding of the current practices and situation in the real world, and the main problems associated with the COTS software selection. Most importantly, these findings can facilitate the development of the new COTS software evaluation and selection framework.

7.2.3 COTS-ESF Development

COTS-ESF is developed to evaluate the COTS software by addressing the non-functional requirements and proposing the required COTS evaluation criteria in order to select the fittest COTS software. The framework also considers the handling of the COTS mismatches. In this context, COTS-ESF is constructed based on three elements: i) evaluation theory, ii) findings of the theoretical study, and iii) findings of the empirical study. The evaluation theory provides six basic components which are evaluation target, evaluation criteria, yardstick, data gathering techniques, synthesis techniques, and evaluation process. These components are adapted in COTS-ESF development to address the COTS software evaluation and selection domain. The findings of the theoretical study are used to identify the required concepts, structures,

and theories such as the MCDM and GA techniques. On the other hand, the findings from the empirical study are used to identify the important COTS evaluation criteria, data gathering techniques, and the main processes and activities of the COTS software evaluation and selection.

COTS-ESF integrates and consolidates the above-mentioned elements in three main processes: planning, preparation, and evaluation and selection. The planning process includes three activities which are defining the evaluation target, forming the evaluation team, and establishing the WBS. The preparation process begins by identifying the functional requirements using the JAD and documents review techniques, searching the COTS alternatives, and defining the yardstick thresholds. Handling the mismatches and making the selection decision are done through two activities, data collection and filtering and decision making, in the evaluation and selection process. Data collection and filtering activity aims to collect the related COTS alternatives data and reducing the number for detail evaluation, while the final score for each COTS alternative can be calculated in order to select the fittest COTS software by the decision making activity. The decision making technique (synthesis technique) in COTS-ESF is established by adapting and combining the AHP and GA techniques. The CEC construction is based on the previous work by Bretoa and Valecillo (2002) and Kalaimagal and Srinivasan (2010/a), which referred to the ISO9126 structure (characteristic, sub-characteristic, and attribute) in providing comprehensive criteria for evaluating the COTS software in four levels of full hierarchy structure.

7.2.4 COTS-ESF Evaluation

The main aim of this study is to evaluate COTS-ESF in supporting the COTS software evaluation and selection in the real life. The evaluation is carried out in two stages, which comprises of verification and validation.

1. **Verification stage**

The framework proposed several essential components for COTS software evaluation and selection such as the CEC and decision making technique. Therefore, the verification stage is required to verify the correctness of their development. The verification process was conducted using the experts review approach coupled with the Delphi technique. The set of criteria used for this purpose consists of comprehensiveness, understandability, accuracy, and coherence. After three rounds of reviewing, the results from the verification process signified that COTS-ESF is acceptable with several modifications. The improved version of COTS-ESF is now able to evaluate the COTS software. More details about these modifications are described section 6.2.

2. **Validation stage**

The overall goal of this stage is to confirm the validity and applicability of COTS-ESF in the real life. To do so, two approaches have been used which are: the case study and yardstick approaches. In regards with the case study, two case studies, Malaysia and Jordan, were chosen to apply COTS-ESF for selecting two different applications. COTS-ESF was evaluated according to various factors of satisfaction such as gain, interface, and task support. Overall, the findings, described in detail in section 6.3.1.4, indicate that COTS-ESF is applicable and effective for evaluating and selecting the COTS software in the real life. In this study, the DM-PT is a useful tool

to support the framework implementation. It was developed to facilitate the evaluation team to apply COTS-ESF in a more systematic and reliable way. The findings indicate that the use of such tool is very helpful in making COTS-ESF implementation easier, efficiently, and more understandable.

To increase the reliability of the validation stage, the yardstick validation approach was used beside other validation methods. It aims to check COTS-ESF validity comparing with other valid models in the field. In this regards, three baseline models have been identified as the basic models in the COTS software evaluation and selection area, which are: OTSO, PORE, and STACE. After that, the criteria of comparison were specified according to the six basic evaluation components that are provided by the Evaluation Theory. COTS-ESF and the other models were investigated based on these components to identify their strengths and weakness. As a result, even though the previous models have contributed to the area of COTS software evaluation and selection by providing basic processes, techniques, and criteria, they have deficiency handling some of the evaluation components such as providing all the required evaluation criteria and accurate decision making technique that addresses COTS mismatches problem. On other hand, the framework has more strengths than previous models such as through addressing non-functional requirements and proposing CEC, and proposing new decision making technique based on integrating AHP and GA to handle COTS mismatches problem. Consequently, COTS-ESF was found valid and useful for evaluating and selecting COTS software comparing with the previous models.

7.3 Research Contributions

As mentioned earlier, several studies have been carried out to construct the new framework. In doing so, this research has several implications on the theory and practice, especially in the area of the COTS software evaluation and selection. The contributions of this research can be divided into two, the main and extra. The main or key contributions include COTS-ESF, CEC, and decision making technique. The extra contributions are theoretical findings, empirical survey findings, data collection and filtering integration, and DM-PT software tool.

The following sections discuss the main research contributions.

7.3.1 COTS-ESF

The main contribution in this research is COTS-ESF for evaluating and selecting the COTS software to address the non-functional requirements and COTS mismatches problems. It was built based on the theoretical and empirical studies in order to address both of the researchers and practitioners perspectives. Most of the previous developed COTS software selection methods have been developed only based on the theory without considering the practice part. This is the reason why most of the frameworks are not applicable in practice.

This research adapted the six evaluation components of the evaluation theory to construct the complete framework. The components and their relationships are presented in a simple and reasonable way which makes its implementation more easy and understandable. This differs to the other previous methods, which did not include those six components. The methods are just focusing either on the evaluation processes, techniques or evaluation criteria.

Moreover, a software tool support was developed to carry out most of COTS-ESF phases and tasks of this in systematic way. This is to ensure that COTS-ESF becomes more systematic and applicable. Using the software tool increases the results reliability especially when it involves a lot of computations. This is essential for the framework to be more acceptable and easy to use. This option (software tool) is missing in most of the existing methods, which make their implementations more difficult especially for those that required many calculations such as the AHP technique.

This framework was developed as a guidance to support and help the developers and organizations to select the appropriate COTS software in a more systematic and repeatable way. The researchers have the opportunity to use this framework as a starting point for future research or to adapt it in other different fields.

7.3.2 CEC

Since most of the previous COTS software evaluation methods and models do not incorporate the non-functional criteria, this research has attempted to include them into the framework. In this context, the CEC was developed to handle the non-functional requirements based on the researchers (theoretical study) and developers' (empirical study) perspectives, and reviewed by two experts.

The five categories of the CEC evaluation criteria include quality, domain, architectural, operational environment, and vendor. By incorporating these categories, the CEC would be more comprehensive compared to the previous works. The comprehensiveness allows estimation to be performed based on various sides or perspectives in ensuring the appropriateness of the COTS software appropriate to

user's needs (e.g. system architecture side and COTS domain side). Moreover, the CEC was developed according to the ISO 9126 model structure that provides a full hierarchy structure of characteristics, sub-characteristics, and attributes. Such hierarchical structure is not found in the previous models and methods.

The CEC is very useful for the practitioners because it provides a variety of evaluation criteria in multi hierarchical levels supported by well-defined metrics. The CEC also facilitates those practitioners to carry out the evaluation process smoothly, easily, and accurately. Apart from that, the wide range of the evaluation criteria in the CEC gives the organizations the opportunity to depend on several qualified evaluators in doing the evaluation and selection process. This would eventually increase the reliability of the evaluation process. The confidence level on the results would also increase as the organization is no longer depending on specific individuals in making the selection decision. This is another important element that is missing in most of the previous methods.

Since it provides a well-defined and completed model that includes many new evaluation criteria, potential researchers can benefit from the CEC model in their future research by integrating it with existing methods. The new model of the COTS selection can also be adapted to other evaluation fields such as in evaluating the OSS components.

7.3.3 The Decision Making Technique

The decision making technique (or synthesis technique) is an essential component for any evaluation process which aims to consolidate and synthesize data in making a right decision. Since the COTS software evaluation and selection is considered as a

multi criteria decision problem, a new decision making technique has been developed by integrating the AHP and GA techniques. The AHP is the common technique used in MCDM, while GA technique is widely used for identifying and analyzing the gaps between the systems and user's requirements.

The AHP was chosen because it uses a hierarchical structure to sort the criteria and the pair-wise comparisons to estimate the relative weights of the criteria. The AHP is known to be the best mechanism to check comparisons consistency and at the same time can reduce bias in decision making. However, the drawback of the AHP is that it cannot handle the gaps that occurred between user's requirements and software features, which would cause inaccurate decision. Therefore, the GA technique was adapted to overcome this limitation. This is necessary because the GA has been accepted as the common technique to identify and analyze such gaps. By adapting and integrating both techniques, an accurate and reliable decision can be delivered by addressing the COTS mismatches, solutions, and constraints such as cost and risk.

The new decision making technique is more reliable and accurate compared to that of the other previous methods, which usually relied on AHP and WSM to consolidate and synthesize the data. The AHP and WSM techniques are not able to address the gaps or mismatches issues. In addition, the proposed technique in this research is supported by the use of a software tool that helps decision makers to implement it in an easy and reliable way.

The research findings indicate that the new decision making technique is accepted by the evaluators in the real life. They considered the technique is helpful in coming up

with accurate, reliable, and effective decision with less time and effort. As for the researchers, the technique can be adapted in other related areas.

The extra contributions of this research are discussed in the following sections.

7.3.4 Theoretical Findings

Most of the research contributions related to the COTS evaluation and selection from the literature were studied and analyzed in order to develop the new framework. Consequently, several processes and activities proposed by previous studies were analyzed and classified accordingly including the related techniques. Among them were the MCDM and GA techniques as well as several related issues. These techniques were analyzed and adapted in this study to achieve the research objectives.

In addition, this research managed to identify, analyze, and compare the existing COTS software evaluation and selection methods in order to address their strengths and weaknesses. Several related challenges and issues were also addressed, discussed and presented. The CBS was defined and handled by studying and analyzing the benefits and risks as well as the main phases of its development process. This related information could be useful and supportive for the conducting further research in the same area or even in the related field of software engineering.

7.3.5 Empirical Survey Findings

The empirical study can be performed to verify, develop new, or extend existing theories, and to improve the current practice. Therefore, in this research, a survey was carried out to identify the important processes, activities and techniques besides the common evaluation criteria for evaluating and selecting the COTS software. In

addition, it was conducted to investigate the current problems and challenges faced by the organizations during the process as well as eliciting the current practice of developing systems using COTS software.

The findings of this survey that comprise of the important processes, activities, techniques, problems, approaches, and common evaluation criteria were delivered based on the developers preferences. These also form the basis of knowledge from the practitioners' perspectives for researchers to develop and improve the existing theories, models, and techniques so that they would become more acceptable in real life. At the same time, these findings could also be used as feedbacks for organizations to not only improve the COTS software selection process but also to encourage the use of a well-defined and systematic method.

7.3.6 Data Collection and Filtering Integration

Generally, for those without the source code, the related data of the COTS software need to be requested from various sources such as the vendors, COTS implementation, and experimental users. In this research, four data sources were applied, which include documentations, experimental users; vendors, and product demonstration. Based on the available data from each source, a set of the COTS software alternatives were estimated and filtered by comparing them with the yardstick thresholds values. As a result, four filtering levels were required for the four sources of data to estimate the COTS software alternatives at each individual level.

The integration between the data collection and filtering is very efficient in decreasing the number of COTS software alternatives. By evaluating the different data at each level, the remainder COTS alternatives in the list are more likely to be the users'

preferences. Besides, the integration also helps in saving the time and efforts rather than to conduct each process separately as in most previous methods.

7.3.7 DM-PT Software Tool

In this thesis, the DM-PT is introduced as a prototyping tool to facilitate and support COTS-ESF implementation. It provides a systematic way to carry out most of the processes and activities in COTS-ESF especially the decision making technique where all the formula and computations are executed systematically. The main benefit of using this tool is the guarantee of having correct and accurate results. This is obvious especially when there are many computation and comparisons such as the CEC weighting task in decision making. Moreover, the DM-PT makes COTS-ESF implementation more efficient in term of time and efforts. This tool is built using the ASP.Net and VB.Net technologies.

7.4 The Research Limitation and Future Work

When this research was completed, several limitations were raised as a result of using a variety of theories, techniques, and methods. On the other hand, COTS-ESF and its findings in this research can be improved and extended by new researches to use in other areas and fields. The studies limitations and future work are discussed in the following sections.

7.4.1 Research Limitations

This research has been conducted to help and support the organizations in developing their systems from COTS software components through proposing COTS-ESF. Since

this research relied on the many methods, techniques, different kinds of data there are several limitations were raised, which are:

Firstly, limitations related to COTS-ESF evaluation phase:

- The validation stage relied on two cases study conducted in two different countries (Malaysia and Jordan). Although COTS-ESF was found effective and applicable in these cases, two cases study are still not enough to fully evaluate the effectiveness of COTS-ESF in order to generalization the results, Thus more cases studies are required for evaluating COTS-ESF and generalizing the results.

Secondly, limitations related the CEC:

- The evaluation criteria in CEC model have been collected from the literature and investigated empirically in practice. Then, the CEC model was reviewed and improved by two experts, and validated through two cases studies to be found acceptable and appropriate for evaluating COTS software. However, the main feedback from the evaluation phase about CEC model is it has many criteria which required effort and time to examine. In this sense, as attempted to overcome this limitation, the CEC model was supported by software tool to help the developers to deal with it.
- Other limitation raised through COTS-ESF evaluation phased is related to the data collection. Using many metrics in CEC model required long time to collect the data about these metrics especially when these metrics are related to vendors and other users of COTS software studies. The required time to collect the data in the two case studies (organization A & B) was more than 50% of all the project time, The response rate in some cases is low and depends on the vendors' cooperation. Therefore, in order to improve the response rate and decrease the data collection time ,this research used remind letters and phone calls with vendors and experimental users of COTS software.

Finally, COTS-ESF is used when the organization is already decided to purchase COTS software rather than developed it in-house. So, this framework cannot be used to make a purchasing versus building decision. In purchasing decision the organization save the cost and efforts by integrating COTS software to develop the system, while in the building decision, the organization uses in-house development to achieve acutely its' requirements.

7.4.2 Future Work

During this research several areas mature for further research and new ideas could be introduced to extend this research. The most pressing ideas for future research are listed as the following:

- This research focuses on selecting the COTS software that fulfills most of the user requirements (single COTS software). The selected COTS software may require customization to meet specific user's needs in building the COTS-solution systems or intermediate systems. COTS-ESF can be adapted not only for selecting multi COTS software but also to facilitate building COTS-aggregate systems.

- Learning from previous selection cases is very important. It helps the evaluation team to learn from the similar previous cases particularly on how well specific vendors supported the software products. Unfortunately, in this research, learning from previous cases is not addressed due to the time limitation. The main reason is that it requires the conducting of a new research to identify the tools, techniques, and methods in the knowledge management. This is necessary in order to manipulate the information and suggest

appropriate criteria weightage, product vendor, and data collection methods based on previous similar projects.

- In this research, the DM-PT software tool is developed to support COTS-ESF implementation. However, this tool is just a prototype version. Therefore, a new improved version is required so that COTS-ESF can be used easier across many domains. The features of this new DM-PT version should enable users to add, remove, and rename the evaluation criteria as well as allowing them to use the diagrams and charts to illustrate the results.

- COTS-ESF, which includes many techniques, processes, tasks, evaluation criteria, and other components, is developed and proposed to evaluate and select the COTS software. The framework or any of its main components such as the CEC or the decision making technique can be adapted by new research in other fields or used to evaluate and select other kind of OTS components such as the OSS (Open Source Software) components.

7.5 Final Conclusion

Due to the increased demands in the last decades, the COTS software has flooded the market by a huge numbers. Consequently, the biggest challenge for organizations is to make appropriate selection that fulfills their requirements. This makes the COTS software evaluation and selection as an important process in developing the CBS. Thus, there are many methods and models proposed for such purpose. Unfortunately, most of these methods and models could not be applied in the real practice and the organizations have no choice rather than using the ad-hoc manner in selecting the COTS software. Usually this ad-hoc mechanism would depend on the manager or

evaluator experience or on particular vendors. The failure of selecting the appropriate COTS software would negatively impact the overall performance of the final system.

The main purpose of developing COTS-ESF is to address the main challenges and problems faced by the practitioner organizations. The main challenge is that the existing methods do not adequately address the following: i) lack of COTS software evaluation criteria due to the inability of handling the non-functional requirements, ii) neglecting other important aspects such as vendor aspects, iii) lack of providing accurate and reliable decision making process, and iv) lack of addressing the COTS mismatches.

This indicates that COTS-ESF has taken into consideration the drawbacks of the previous methods and models by proposing the CEC in addressing the non-functional requirements. This is done by providing the required evaluation criteria for evaluating the COTS software. In addition, the COTS mismatches problem is also put forward by proposing the decision making technique, which combine both the AHP and GA techniques in order to provide accurate decision for the selection purposes. As a result, this framework does provide guideline for future theoretical research direction, as well as being practical tool, usable in the real contexts.

REFERENCES

Acuña, S. T., Antonio, A. D., Ferré, X., Lopez, M., & Mate, L. (2001). The software process: Modelling, evaluation and improvement. In S. K. Chang (Eds.), *Handbook of Software Engineering and Knowledge Engineering* (Vol. 1, pp. 193-237). Singapore: World Scientific Publishing Company Incorporated.

Akhavi, F., & Hayes, C. (2003). A comparison of two multi-criteria decision-making techniques. *In proceeding of the Systems, Man and Cybernetics 2003, IEEE International Conference held on 5-8 Oct. 2003 at the Washington DC* (Vol. 1, pp. 956-961). USA: IEEE.

Alanbay, O. (2005). ERP selection using expert choice software. *In Proceeding of the ISAHP held on 8-10 July 2005 at the Honolulu, Hawaii* (47). USA: Elmhurst College.

Albert, C., & Brownsword, L. (2002). *Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview* (CMU/SEI-2002-TR-009). Software Engineering Institute, Carnegie Mellon University, Pennsylvania, United States.

Alghamdi, A. (2007). An Empirical Process for Evaluating and Selecting AWEM Environments:" Evaluation Stage". *Comp. & Info. Sci.*, *19*(1), 17-37. Retrieved from http://ccisj.ksu.edu.sa/ccisj/CCIS/

Alvaro, A., Almeida, E. S., Vasconcelos, A. M. L., & Meira, S. R. L. (2005, Aug). *Towards a software component quality model*. Paper presented at the 5th International Conference on Quality Software (QSIC), Work in Progress Session, Porto, Portugal.

Alvaro, A., de Almeida, E. S., & Meira, S. L. (2006). A Software Component Quality Model: A Preliminary Evaluation. *In proceeding of the Software Engineering and Advanced Applications (SEAA '06) 32nd EUROMICRO Conference held on 29 Aug. – 1 Spt. 2006 at the Cavtat, Dubrovnik* (pp. 28-37). Croatia: IEEE.

Alvaro, A., Santana de Almeida, E., & Romero de Lemos Meira, S. (2010). A software component quality framework. *ACM SIGSOFT Software Engineering Notes*, *35*(1), 1-18. doi: 10.1145/1668862.1668863

Alves, C., & Castro, J. (2001, Oct). *CRE: A systematic method for COTS components selection*. Paper presented at the XV Brazilian Symposium on Software Engineering (SBES), Rio de Janeiro, Brazil.

Alves, C., & Finkelstein, A. (2003). Investigating conflicts in COTS decision-making. *International Journal of Software Engineering and Knowledge Engineering*, *13*(5), 473-493. doi: 10.1142/S0218194003001408

Alves, C., Castro, J., & Alencar, F. (2000, July). *Requirements Engineering for COTS Selection*. Paper presented at the 3th Workshop on Requirements Engineering, Rio de Janeiro, Brazil. Retrieved from http://wer.inf.puc-rio.br/wer01/NFu-Req-1.pdf

Alves, C., Franch, X., Carvallo, J., & Finkelstein, A. (2005). Using Goals and Quality Models to Support the Matching Analysis during COTS Selection. In X. Franch & D. Port (Eds.), *COTS-Based Software Systems* (Vol. 3412, pp. 146-156). Bilbao, Spain: Springer Berlin Heidelberg.

Alves, C., Pinto Filho, J. B., & Castro, J. (2001, Nov). *Analysing the tradeoffs among requirements, architectures and COTS components*. Centro de Informática, Universidade Federal de Pernambuco Recife, Pernambuco. Retrieved on 26 Jul. 2009 from http://www.inf.puc-rio.br/~wer/WERpapers/artigos/artigos_WER01/alves.pdf

Ariff, H., Salit, M. S., Ismail, N., & Nukman, Y. (2012). Use of Analytical Hierarchy Process (AHP) for selecting the best design concept. *Jurnal Teknologi*, *49*, 1–18. doi: 10.11113/jt.v49.188

Asghar, S., & Umar, M. (2010). Requirement Engineering Challenges in Development of Software Applications and Selection of Customer-off-the-Shelf (COTS) Components. *International Journal of Software Engineering (IJSE)*, *1*(2), 32. Retrieved from http://cscjournals.org/csc/manuscript/Journals/IJSE/volume1/Issue2/IJSE-7.pdf

Assadi, P., & Sowlati, T. (2009). Design and manufacturing software selection in the wood industry using analytic hierarchy process. *International Journal of Business Innovation and Research*, *3*(2), 182-198. doi: 10.1504/IJBIR.2009.022754

Avizienis, A., Laprie, J. C., Randell, B., & Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. Dependable and Secure Computing, IEEE Transactions, *1*(1), 11-33. doi: 10.1109/tdsc.2004.2

Ayala, C. (2008). *Systematic Construction of Goal- Oriented COTS Taxonomies*. Unpublished Master Thesis, Departament of Llenguatges i Sistemes Informàtics,Universitat Politècnica de Catalunya, Barcelona, Spain.

Ayala, C., Hauge, Ø., Conradi, R., Franch, X., & Li, J. (2011). Selection of third party software in Off-The-Shelf-based software development—An interview study with industrial practitioners. *Journal of Systems and Software, 84*(4), 620-637. Retrieved from http://www.sciencedirect.com/science/article/pii/S0164121210002864

Bachmeyer, K. (2004). *What Does COTS Really Mean?*. Retrieved on 13 May, 2009 from http://www.gisdevelopment.net/proceedings/gita/2005/papers/58.pdf

Bailey, K. (2008). *Methods of Social Research* (4th Edition). New York: Free Press.

Basili, V., & Boehm, B. (2001). COTS-Based Systems Top 10 List. *IEEE Computer Society Press, 34*(5), 91 - 93. doi: 10.1109/2.920618

Becker, C., & Rauber, A. (2009). Requirements modelling and evaluation for digital preservation: a COTS selection method based on controlled experimentation. *In Proceedings of the 2009 ACM symposium on Applied Computing at the Honolulu University* (pp. 401-402). Hawaii, USA: ACM.

Behkamal, B., Kahani, M., & Akbari, M. K. (2009). Customizing ISO 9126 quality model for evaluation of B2B applications. *Information and software Technology, 51*(3), 599-609. doi: 10.1016/j.infsof.2008.08.001

Beil, D. (2009). *Supplier Selection*. Wiley Encyclopedia of Operations Research and Management Science, University of Michigan, Michigan, United States. Retrieved on 22 Jun 2010 from http://www-personal.umich.edu/~dbeil/Supplier_Selection_Beil-EORMS.pdf

Berch, D. B. (2003, July). Gap *Analysis: Bridging the space between what we know and what we want to know*. Paper presented at the National Research Council's Workshop on

Understanding and Promoting Knowledge Accumulation in Education: Tools and Strategies for Educational Research, Washington, USA.

Bertoa, M., & Vallecillo, A. (2002, Jun). *Quality attributes for COTS components*. Paper presented at the Proceedings of the 6th International Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'2002), Malagas, Spain.

Beus-Dukic, L. (2000). Non-functional requirements for COTS software components. *In Proceedings of the COTS workshop: Continuing Collaborations for Successful COTS Development (ICSE2000) held on 4-5 June 2000 at the University of Limerick* (pp. 4-5). Limerick, Ireland: ACM.

Bharathi, V., Vaidya, O., & Parikh, S. (2012). Prioritizing and Ranking Critical Success Factors for ERP Adoption in SMEs. *AIMS International Journal of Management, 6*(1), 23-40. Retrieved from http://ssrn.com/abstract=2188829

Bhuta, J., & Boehm, B. (2005). A Method for Compatible COTS Component Selection. In X. Franch & D. Port (Eds.), *COTS-Based Software Systems* (Vol. 3412, pp. 132-143). Bilbao, Spain: Springer Berlin Heidelberg.

Bhuta, J., & Boehm, B. (2007). Attribute-based cots product interoperability assessment. *In Proceeding of the 6th International IEEE Conference in Commercial-off-the-Shelf (COTS)-Based Software Systems ICCBSS'07 held on 26 Feb. - 2 March 2007 at the Banff, Albt.* (pp. 163-171). Alberta, Canada: IEEE.

Birisci, S., Metin, M., & Karakas, M. (2009). Prospective elementary teachers' attitudes toward computer and Internet use: A sample from Turkey. *World Applied Sciences Journal, 6*(10), 1433-1440. Retrieved from http://idosi.org/wasj/wasj6(10)/20.pdf

Boehm, B., & Abts, C. (1999). COTS integration: Plug and Pray?. *Computer, 32*(1), 135-138. doi: 10.1109/2.738311

Boehm, B., Port, D., Yang, Y., & Bhuta, J. (2003). Not All CBS Are Created Equally: COTS-Intensive Project Types. In H. Erdogmus & T. Weng (Eds.), *COTS-Based Software Systems* (Vol. 2580, pp. 36-50). Ottawa, Canada: Springer Berlin Heidelberg.

Botella, P., Burgués, X., Carvallo, J., Franch, X., Grau, G., Marco, J., et al. (2004, Jan). *ISO/IEC 9126 in practice: what do we need to know?*. Paper presented at the Proceedings of the 1st Software Measurement European Forum (SMEF), Roma.

Bordley, R. F. (2001). Integrating gap analysis and utility theory in service research. *Journal of Service Research, 3*(4), 300-309. doi: 10.1177/109467050134003

Brownsword, L., Oberndorf, T., & Sledge, C. (2000). Developing new processes for COTS-based systems. *IEEE software, 17*(4), 48-55. doi: 10.1109/52.854068

Bruseberg, A. (2006). The design of complete systems: Providing human factors guidance for COTS acquisition. *Reliability Engineering & System Safety, 91*(12), 1554-1565. doi:10.1016/j.ress.2006.01.016

Bryman, A., & Bell, E. (2007). *Business research methods*. USA: Oxford University Press.

Buchanan, J., & Vanderpooten, D. (2007). Ranking projects for an electricity utility using ELECTRE III. *International Transactions in Operational Research, 14*(4), 309-323. doi: 10.1111/j.1475-3995.2007.00589.x.

Burgués, X., Estay, C., Franch, X., Pastor, J., & Quer, C. (2002). Combined selection of COTS components. *COTS-Based Software Systems, 2255*(5), 54-64. doi: 10.1007/3-540-45588-4_6.

Cai, L., Xie, X., & Huang, S. (2011). Software Quality Model Development-An Introduction. *Energy Procedia, 13*, 8749-8758.

Callagy, C. (2007). *Commercial Off-The-Shelf Software: Using Purchasing Portfolio Management to Gain Competitive Advantage*. Unpublished Doctoral Thesis, Department of Management, Faculty of Commerce, National University of Ireland, Galway, Ireland.

Cangussu, J., Cooper, K., & Wong, E. (2006). Multi Criteria Selection of Components Using the Analytic Hierarchy Process. In I. Gorton, G. Heineman, I. Crnković, H. Schmidt, J. Stafford, C. Szyperski & K. Wallnau (Eds.), *Component-Based Software Engineering* (Vol. 4063, pp. 67-81). Västerås, Sweden: Springer Berlin Heidelberg.

Carney, D. (1997, Jun). *Assembling large systems from COTS components: opportunities, cautions, and complexities*. Paper presented at the SEI Monographs on Use of Commercial Software in Government Systems. Pittsburgh, PA, US. Retrieved from http://www.sei.cmu.edu/library/assets/assemblingsystems.pdf

Carney, D., Hissam, S., & Plakosh, D. (2000). Complex COTS-based software systems: practical steps for their maintenance. *Journal of Software Maintenance: Research and Practice, 12*(6), 357-376. doi: 10.1002/1096-908X(200011/12)12:6<357::AID-SMR219>3.0.CO;2-L

Carson, J. S., II. (2002, Dec). Model verification and validation. *Paper presented In Proceeding of the 2002 Winter Simulation Conference, held on 8-11 Dec. 2002 at the Marietta, GA* (pp. 52-58 Vol.1). USA: IEEE Computer Society.

Carvallo, J. P., & Franch, X. (2006). Extending the ISO/IEC 9126-1 quality model with non-technical factors for COTS components selection. *Paper Proceedings in the 2006 international workshop of Software quality held on 20-28 May 2006 at the Shanghai International Convention Center* (pp.9-14). Shanghai, China: ACM.

Carvallo, J. P., Franch, X., Grau, G., & Quer, C. (2004, September). COSTUME: a method for building quality models for composite COTS-based software systems. *In Proceedings of the 4th International Conference on Quality Software, QSIC held on 8-9 Sept. 2004 at the Braunschweig* (pp. 214-221). Germany: IEEE Computer Society.

Cechich, A., & Piattini, M. (2007). A Six Sigma-Based Process to Improve COTS Component Filtering. *Journal of Research and Practice in Information Technology, 39*(4), 245-272. Retrieved from http://pdf.aminer.org/000/366/102/quantifying_cots_component_functional_adaptation.pdf

Cechich, A., & Taryano, K. (2003, Oct). *Selecting COTS components: a comparative study on E-Payment systems*. Paper presented in IX Congreso Argentino de Ciencias de la

Computación, CAECE University, Mar del Plata, Argentina. Retrieved from http://sedici.unlp.edu.ar:8080/handle/10915/22582

Charpentier, R., & Salois, M. (2000). Detection of Malicious Code in COTS Software via Certifying Compilers. *In proceeding of the Commercial Off-The-Shelf Products in Defence Applications "The Ruthless Pursuit of COTS" held on 3-5 Apr 2000 at the Neuilly-sur-Seine Cedex, France* (pp.1-8). France: RTO.

Chen McCain, S.-L., Jang, S., & Hu, C. (2005). Service quality gap analysis toward customer loyalty: practical guidelines for casino hotels. *International Journal of Hospitality Management, 24*(3), 465-472. doi://dx.doi.org/10.1016/j.ijhm.2004.09.005

Chung, L., & Cooper, K. (2004, Jun). *COTS-aware requirements engineering and software architecting*. Paper presented at the Proceeding of Software Engineering Research and Practice Conference (SERP), Las Vegas, NE, USA. Retrieved from http://pdf.aminer.org/000/585/051/cots_aware_requirements_engineering_and_software_architecting.pdf

Chung, L., & Prado Leite, J. (2009). On Non-Functional Requirements in Software Engineering. In A. Borgida, V. Chaudhri, P. Giorgini & E. Yu (Eds.), *Conceptual Modeling: Foundations and Applications* (Vol. 5600, pp. 363-379). Canada: Springer Berlin Heidelberg.

Chung, L., Cooper, K., & Huynh, D. (2001, Oct). *COTS-aware requirements engineering Technique*. Paper presented at the Proceedings of the 2001 Workshop on Embedded Software Technology (WEST01), Tahoe City, CA, USA. Retrieved on 22 Mar. 2009 from http://cs.mvnu.edu/twiki/pub/Main/SoftwareEngineering2010/COCOTS-Aware.pdf

Clough, A. (2003). *Commercial Off-the-shelf (COTS) Hardware and Software for Train Control Applications: System Safety Considerations* (Report.No. DOT-VNTSC-FRA-02-01,). National Technical Information Service, USA.

Comella-Dorda, S., Dean, J., Morris, E., & Oberndorf, P. (2002). A Process for COTS Software Product Evaluation. In J. Dean & A. Gravel (Eds.), *COTS-Based Software Systems* (Vol. 2255, pp. 86-96).Orlando, Florida: Springer Berlin Heidelberg.

Comella-Dorda, S., Dean, J. C., Lewis, G., Morris, E. J., Oberndorf, P. A., & Harper, E. (2004). *A process for COTS software product evaluation* (Technical Report CMU/SEI-2003-TR-017). Software Engineering Institute, Pittsburgh: Carnegie Mellon University, Pennsylvania, USA.

Cortellessa, V., Crnkovic, I., Marinelli, F., & Potena, P. (2007). Driving the selection of COTS components on the basis of system requirements. *In Proceedings of the 22th IEEE/ACM international conference on Automated software engineering held on 5-9 November 2007 at the Atlanta, Georgia* (pp. 413-416). Georgia, USA: ACM.

Couts, C. T., & Gerdes, P. F. (2010). Integrating COTS Software: Lessons from a Large Healthcare Organization. *IT Professional, 12*(2), 50-58. doi: 10.1109/mitp.2010.59

Crnkovic, I., Larsson, M., & Preiss, O. (2005). Concerning predictability in dependable component-based systems: Classification of quality attributes. *Architecting Dependable Systems III, Lecture Notes in Computer Science, 3549*, 257-278. Springer Berlin Heidelberg. Retrieved from

http://link.springer.com/chapter/10.1007%2F11556169_12?LI=true

Dagdeviren, H., Juric, R., & Kassana, T. A. (2005). An exploratory study for effective COTS and OSS product marketing. *In Proceeding of the 27th International Conference on Information Technology Interfaces held on 20-23 June 2005 at the Cavtat* (pp. 644–649). Croatia: IEEE.

Daim, T. U., Udbye, A., & Balasubramanian, A. (2013). Use of analytic hierarchy process (AHP) for selection of 3PL providers. *Journal of Manufacturing Technology Management, 24*(1), 28-51. doi: 10.1108/17410381311287472

Dalkey, N., & Helmer, O. (1963). An experimental application of the Delphi method to the use of experts. *Management Science, 9*(3), 458-467. doi: 10.1287/mnsc.9.3.458

Dean, J., & Gravel, A. (Eds.). (2002). COTS-Based Software Systems*: Proceedings of 1th International Conference on COTS-Based Software Systems (ICCBSS 2002),* Orlando University, FL, Feb 2002, USA: Springer.

Dolan, J. G. (2010). Multi-criteria clinical decision support: A primer on the use of multiple criteria decision making methods to promote evidence-based, patient-centered healthcare. *The patient, 3*(4), 229. doi: 10.2165/11539470-000000000-00000

Dustin, E., Rashka, J., & McDiarmid, D. (2002). *Quality web systems: performance, security, and usability*. Boston, USA: Addison-Wesley Longman Publishing.

Dzever, S., Merdji, M., & Saives, A.-L. (2001). Purchase decision making and buyer-seller relationship development in the French food processing industry. *Supply Chain Management: An International Journal, 6*(5), 216-229. doi: 10.1108/13598540110407769

Egyed, A., & Balzer, R. (2006). Integrating cots software into systems through instrumentation and reasoning. *Automated Software Engineering, 13*(1), 41-64. doi: 10.1007/s10515-006-5466-4

Elanchezhian, C., Ramnath, B. V., & Kesavan, R. (2010). Vendor selection using analytical hierarchy process in supply chain management. *Journal of Engineering Research and Studies, 1*(1), 118-127. Retrieved from http://www.technicaljournalsonline.com

Eldrandaly, K. (2007). An Intelligent MCDM Approach for Selecting the Suitable Expert System Building Tool. *the International Arab Journal of Information Technology (IAJIT), 4*(4), 365-371. Retrieved from http://history.kau.edu.sa/Files/195/Researches/58680_28927.pdf

Erol, I., & Ferrell Jr, W. G. (2003). A methodology for selection problems with multiple, conflicting objectives and both qualitative and quantitative criteria. *International Journal of Production Economics, 86*(3), 187-199. doi: 10.1016/S0925-5273(03)00049-5

Fahmi, S. A., & Choi, H. J. (2009). A study on software component selection methods. *In proceeding of the 11th International Conference In Advanced Communication Technology (ICACT 2009) held on 15-18 Feb. 2009 at the Phoenix Park* (Vol. 1, pp. 288-292). Dublin, Ireland: IEEE.

Falessi, D., Cantone, G., Kazman, R., & Kruchten, P. (2011). Decision-making techniques for software architecture design: a comparative survey. *ACM Computing Surveys (CSUR), 43*(4), 33. doi: 10.1145/1978802.1978812.

Fan, C. K., & Cheng, S. W. (2009). Using analytic hierarchy process method and technique for order preference by similarity to ideal solution to evaluate curriculum in department of risk management and insurance. *Journal of Science in Society, 19*(1), 1-8. Retrieved from http://www.krepublishers.com/02-Journals/JSS/JSS-19-0-000-09-Web/JSS-19-1-000-09-Abst-PDF

Fang, G., Lin, J., Chin, K., & Lee, C. (2010). Software Integration for Applications with Audio/Video Stream. *International Journal of Innovative Computing, Information and Control, 6*(3), 1421–1433. Retrieved from http://www.ijicic.org/ihmsp08-21-1.pdf

Felice, F., & Petrillo, A. (2012). Hierarchical Model to Optimize Performance in Logistics Policies: Multi Attribute Analysis. *Procedia - Social and Behavioral Sciences, 58*(0), 1555-1564. doi: http://dx.doi.org/10.1016/j.sbspro.2012.09.1142

Fisher, C. M. (2007). *Researching and writing a dissertation: a guidebook for business students*. England: Prentice Hall.

Fox, R. J. 2010. *Non probability Sampling*. USA: Wiley International Encyclopedia of Marketing.

Franch, X., & Carvallo, J. P. (2003). Using quality models in software package selection. *Software, IEEE, 20*(1), 34-41. doi: 10.1109/ms.2003.1159027

Fröberg, J. (2000). *Software components and COTS in software system development*. Faculty of Information System and Technology, Mälardalen University, Västerås, Sweden.

Galorath, D. (2005). *Software Reuse and Commercial Off-the-Shelf Software*. Galorath Incorporation, El Segundo, CA, USA. retrieved on 4 February 2009 from http://www.compaid.com/caiinternet/ezine/galorath-reuse.pdf

Gay, L. R., Mills, G. E., & Airasian, P. (2006). *Educational research: Competencies for analysis and application* (8th Edition). Upper Saddle River, NJ: Pearson Merrill Prentice Hall.

Gayen, T., & Misra, R. (2009). Reliability Assessment of Elementary COTS Software Component. *International Journal of Recent Trends in Engineering, 1*(2), 196-200. Retrieved from http://www.ijrte.academypublisher.com/vol01/no02/ijrte0102196200.pdf

George, B., Fleurquin, R., & Sadou, S. (2008). A Component Selection Framework for COTS Libraries. In M. V. Chaudron, C. Szyperski & R. Reussner (Eds.), *Component-Based Software Engineering* (Vol. 5282, pp. 286-301). Karlsruhe, Germany: Springer Berlin Heidelberg.

Gerea, M. (2006). *Selection and Evaluation of Open Source Components*. Unpublished Doctoral Thesis. Department of Computer and Information Science. Norwegian University of Science and Technology (NTNU), Trondheim., Norway.

Gill, N. S. (2006). Importance of software component characterization for better software reusability. SIGSOFT Softw. *Eng. Notes, 31*(1), 1-3. doi: 10.1145/1108768.1108771

Glinz, M. (2007). On non-functional requirements. *In proceeding of the 15th IEEE International in Requirements Engineering Conference (RE'07) held on 15-19 October 2007 at the Habitat World Convention Centre, Delhi* (pp. 21-26). India: IEEE.

Gomes, C. F. S., Nunes, K. R. A., Helena Xavier, L., Cardoso, R., & Valle, R. (2008). Multi-criteria decision making applied to waste recycling in Brazil. *Omega, 36*(3), 395-404. Retrieved from http://www.sciencedirect.com/science/article/pii/S030504830600123X

Goswami, V., & Acharya, Y. (2009, Nov). *Method for Reliability Estimation of COTS Components based Software Systems*. Paper presented at the proceedings of 20th International Symposium on Software Reliability Engineering*, ISSRE*, Bengaluru-Mysuru, India.

Grau, G., Carvallo, J. P., Franch, X., & Quer, C. (2004). DesCOTS: a software system for selecting COTS components. *In Proceedings of the 30th Euromicro Conference held on 31 Aug.-3 Sept. 2004 at the Rennes, France* (pp. 118-126).France: IEEE.

Gregor, S., Hutson, J., & Oresky, C. (2002). Storyboard Process to Assist in Requirements Verification and Adaptation to Capabilities Inherent in COTS. In J. Dean & A. Gravel (Eds.), *COTS-Based Software Systems* (Vol. 2255, pp. 132-141). FL, USA: Springer Berlin Heidelberg.

Guerra, PA de C., Rubira, C., Romanovsky, A., & de Lemos, R. (2004) A Dependable Architecture for COTS-based Software Systems using Protective Wrappers. In: de Lemos R, Gacek C, Romanovsky A (eds.), *Architecting Dependable Systems II, LNCS* (Vol. 3069*,* pp. 144–166). Portland, USA: Springer Berlin Heidelberg.

Gulen, K. G. (2007). Supplier selection and outsourcing strategies in Supply Chain Management. *Journal of aeronautics and space technologies, 3*(2), 1-6. Retrieved from http://www.bs2011.hho.edu.tr/web_eng/egitim_programlari/lisansustu_egitim/Huten%20Dergi/2007Temmuz/4_GULEN.pdf

Gupta, P., Mehlawat, M. K., & Verma, S. (2012). COTS selection using fuzzy interactive approach. *Optimization Letters, 6*(2), 273-289. doi: 10.1007/s11590-010-0243-5

Gupta, P., Verma, S., & Mehlawat, M. (2012). Optimization Model of COTS Selection Based on Cohesion and Coupling for Modular Software Systems under Multiple Applications Environment. In B. Murgante, O. Gervasi, S. Misra, N. Nedjah, A. C. Rocha, D. Taniar & B. Apduhan (Eds.), *Computational Science and Its Applications – ICCSA 2012* (Vol. 7335, pp. 87-102). Bahia, Brazil: Springer Berlin Heidelberg.

Hallowell, M. R., & Gambatese, J. A. (2010). Qualitative research: application of the Delphi method to CEM research. *Journal of construction engineering and management, 136*(1), 99-107. doi: 10.1061/_ASCE_CO.1943-7862.0000137

Hämäläinen, R., Kettunen, E., Marttunen, M., & Ehtamo, H. (2001). Evaluating a framework for multi-stakeholder decision support in water resources management. *Group decision and negotiation, 10*(4), 331-353. doi: 10.1023/A:1011207207809

Herraiz, I., Shihab, E., Nguyen, T. H., & Hassan, A. E. (2011). Impact of installation counts on perceived quality: A case study on debian. *In Proceeding of the 18th Working Conference on Reverse Engineering (WCRE) held on 17-20 Oct 2011at the Limerick, Ireland* (pp.219-228). Ireland: IEEE.

Hill, R., Wang, J., & Nahrstedt, K. (2004). Quantifying non-functional requirements: a process oriented approach. *In Proceedings of the 12th IEEE International Conference of Requirements Engineering held on 6-11 Sept. 2004 at the Kyoto, Japan* (pp. 352-353). Japan: IEEE.

Hopkins, J. (2000). Component primer. *Communications of the ACM, 43*(10), 27-30. doi: 10.1145/352183.352198

Hsiao, S.-W. (2002). Concurrent design method for developing a new product. *International Journal of Industrial Ergonomics, 29*(1), 41-55. doi: http://dx.doi.org/10.1016/S0169-8141(01)00048-8

Hsiao, W. H., Chang, T. S., Huang, M. S., & Chen, Y. C. (2011). Selection criteria of recruitment for information systems employees: Using the analytic hierarchy process (AHP) method. *African Journal of Business Management, 5*(15), 6201-6209. doi: 10.5897/AJBM11.185

Hsu, C.-C., & Sandford, B. A. (2007). The Delphi technique: Making sense of consensus. Practical Assessment, *Research & Evaluation, 12*(10), 1-8. Retrieved from http://pareonline.net/getvn.asp?v=12&n=10

Huang, J., Wu-chi, F., & Wu-chang, F. (2006). Cascades: scalable, flexible and composable middleware for multi-modal sensor networking applications. *In Proceeding of SPIE 6071, Multimedia Computing and Networking held on 15-18 January 2006 at the San Jose State University* (Vol. 6071). CA, USA: International Society for Optics and Photonics.

Ibrahim, H., Elamy, A.-H. H., Far, B. H., & Eberlein, A. (2011). UnHOS: A Method for Uncertainty Handling in Commercial Off-The-Shelf (COTS) Selection. *International Journal of Energy, Information and Communications, 2*(3), 21-48. Retrieved from http://www.sersc.org/journals/IJEIC/vol2_Is3/3.pdf

Ibrahim, H., Far, B. H., Eberlein, A., & Daradkeh, Y. (2009). Uncertainty management in software engineering: Past, present, and future. In Proceeding of the *Canadian Conference on the Electrical and Computer Engineering (CCECE'09) held on 3-6 May 2009 at the St. John's, NL* (pp. 7-12). Canada: IEE.

Ismail, W., Abedlazeez, N., & Hussin, Z. (2011). Epistemological Beliefs of Students at High Schools: A Survey Study in Malaysia. OIDA *International Journal of Sustainable Development, 2*(08), 39-46. Retrieved from http://ssrn.com/abstract=1974094

ISO. (2011). *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models*, International Organization for Standardization, J.S. ISO/IEC 25010:2011.

Jackson, M. J., Helms, M. M., & Ahmadi, M. (2011). Quality as a gap analysis of college students' expectations. *Quality Assurance in Education, 19*(4), 392-412. doi: 10.1108/09684881111170096

Jadhav, A., & Sonar, R. (2008). A hybrid system for selection of the software packages. *In Proceeding of 1th International Conference Emerging Trends in Engineering and Technology (ICETET'08) held on 16-18 July 2008 at the University of Nagpur* (pp. 337-342). Nagpur, Maharashtra: IEEE.

Jadhav, A., & Sonar, R. (2009). An Integrated Rule-Based and Case-Based Reasoning Approach for Selection of the Software Packages. *Springer Berlin Heidelberg, 31*(5), 280-291. doi: 10.1007/978-3-642-00405-6_30

Jadhav, A. S., & Sonar, R. M. (2011). Framework for evaluation and selection of the software packages: A hybrid knowledge based system approach. *Journal of Systems and Software, 84*(8), 1394-1407. doi: 10.1016/j.jss.2011.03.034

Javed, Z., Sattar, A. R., & Faridi, M. S. (2012). Unsolved Tricky Issues on COTS Selection and Evaluation. *Global Journal of Computer Science and Technology, 12*(10-D). retrieved from http://computerresearch.org/stpr/index.php/gjcst/article/viewArticle/1181.

Jha, P. C., & Bali, S. (2012). Optimal COTS selection for fault tolerant software system with mandatory redundancy in critical modules using consensus recovery block scheme under fuzzy environment. *In Proceeding of Recent Advances in Computing and Software Systems International Conference (RACSS) held on 25-27April 2012 at the University of Madras, Chennai* (pp. 273-280). India: IEEE.

Jingyue, L., Conradi, R., Bunse, C., Torchiano, M., Slyngstad, O., & Morisio, M. (2009). Development with Off-the-Shelf Components: 10 Facts. Software, *IEEE, 26*(2), 80-87. doi: 10.1109/ms.2009.33

Johar, A. K., & Goel, S. (2011). COTS Components Usage Risks in Component Based Software Development. *International Journal of Information Technology, 4*(2), 573-575. Retrieved from http://www.csjournals.com/IJITKM/PDF%204-2/Article_45.pdf

Kalaimagal, S., & Srinivasan, R. (2008). A retrospective on software component quality models. *SIGSOFT Software. Eng. Notes, 33*(6), 1-10. doi: 10.1145/1449603.1449611

Kalaimagal, S., & Srinivasan, R. (2010/a). Q FACTO 10-A commercial off-the-shelf component quality model proposal. *J. Software Eng, 4*(1), 1-15. doi: 10.3923/jse.2010.1.15

Kalaimagal, S., & Srinivasan, R. (2010/b). Q'Facto 12: an improved quality model for COTS components. *ACM SIGSOFT Software Engineering Notes, 35*(2), 1-4. doi: 10.1145/1734103.1734116

Kaur, A., & Mann, K. S. (2010). Component Selection for Component Based Software Engineering. *International Journal of Computer Applications IJCA, 2*(1), 109-114.retrieved from www.psu.edu

Keil, M., & Tiwana, A. (2005). Beyond cost: the drivers of COTS application value. *IEEE software, 22*(3), 64-69. doi: 10.1109/MS.2005.58

Kesseler, E. (2008). Assessing COTS software in a certifiable safety-critical domain. *Information Systems Journal, 18*(3), 299-324. doi:10.1111/j.1365-2575.2007.00257.x

Kim, S., & Park, J. (2003). C-QM: a practical quality model for evaluating cots components. *In Proceedings of the International Association of Science and Technology for Development (IASTED) International Conference on Software Engineering held on 10-13 February 2003 at the Innsbruck* (pp. 991–996). Austria: ACTA Press.

Kirakowski, J. (2000). *Questionnaires in usability engineering: a list of frequently asked questions* (3rd edition.). Cork, Ireland: Human Factors Research Group.

Kitchenham, B. A., & Pickard, L. M. (1998). Evaluating software engineering methods and tools: part 9: quantitative case study methodology. *SIGSOFT Softw. Eng. Notes, 23*(1), 24-26. doi: 10.1145/272263.272268

Kiv, S., Kolp, M., Filipe, J., Fred, A. L. N., & Sharp, B. (2010). A process for COTS-selection and mismatches handling-a goal-driven approach. *In proceeding of the 2th International Conference on Agents and Artificial Intelligence, ICAART2010 held on 22-24 Jan. 2010 at the Valencia* (pp.98-106). Spain: Hogeschool-Universiteit Brussel (HUB).

Kontio, H. (2008, April). *Current trends in software industry: COTS Integration*. Retrieved on 19 May 2009 from http://www.cs.helsinki.fi/u/paakki/Kontio.pdf

Kontio, J. (1995). *OSTO: A Systematic Process for Reusable Software Component Selection*. (Technical Rep. No. CSTR-3478). University of Maryland at College Park, Maryland, USA. Retrieved from http://dl.acm.org/

Kontio, J. (1996). A case study in applying a systematic method for COTS selection. *In Proceedings of the 18th International Conference in Software Engineering held on 25-29 Mar 1996 at the Berlin* (pp. 201-209). Berlin, Germany: IEEE.

Kumar, R., & Singh, M. K. (2012). A Literature Survey on black box testing in component based software engineering. *International Journal, 2*(3), 420-423. Retrieved from www.ijarcsse.com

Kumar, S., & Suri, P. (2012). Effort Distribution in COTS Components Integration: A Simulation Based Approach. *Global Journal of Computer Science and Technology, 12*(3). Retrieved from http://computerresearch.org/stpr/index.php/gjcst/article/view/1011/894

Kumari, U., & Upadhyaya, S. (2011). An Interface Complexity Measure for Component-based Software Systems. *International Journal of Computer Applications, 36*(1), 46-52. Retrieved from http://research.ijcaonline.org/volume36/number1/pxc3976247.pdf

Kunda, D. (2002). *A social-technical approach to selecting software supporting COTS-Based Systems*, Unpublished Doctoral Thesis, Department of Computer Science, University of York, York, UK. .

Kunda, D. (2003). STACE: social technical approach to COTS software evaluation. *Lecture Notes in Computer Science, 2693*, 64-84. doi: 10.1007/978-3-540-45064-1_4

Kunda, D., & Brooks, L. (1999). Applying social-technical approach for COTS selection. *In Proceedings of the 4th UKAIS Conference held on 7-9 April 1999 at the University of York* (pp. 552-565). York, UK: McGraw-Hill.

Kvale, A., Li, J., & Conradi, R. (2005). A case study on building COTS-based system using aspect-oriented programming. *In Proceedings of the 20th Annual ACM Symposium on Applied Computing held on 13-17 Mar. 2005 at the New Mexico Institute of Mining and Technology, Socorro, NM, Santa Fe* (p. 1491-1498). New Mexico, USA: ACM.

Land, R., & Blankers, L. (2007). *Classifying and Consolidating Software Component Selection Methods*. (Report-MRTC-218/2007-1-SE), Mälardalen Real-Time Research Centre, Mälardalen University, Sweden. Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:mdh:diva-7091

Land, R., Blankers, L., Chaudron, M., & Crnković, I. (2008). COTS Selection Best Practices in Literature and in Industry. In H. Mei (Ed.), *High Confidence Software Reuse in Large Systems* (Vol. 5030, pp. 100-111). Parma, Italy: Springer Berlin Heidelberg.

Land, R., Sundmark, D., Lüders, F., Krasteva, I., & Causevic, A. (2009). Reuse with Software Components - A Survey of Industrial State of Practice. In S. Edwards & G. Kulczycki (Eds.), *Formal Foundations of Reuse and Domain Engineering* (Vol. 5791, pp. 150-159). Falls Church, VA, USA: Springer Berlin Heidelberg.

Lee, Y. W., Strong, D. M., Kahn, B. K., & Wang, R. Y. (2002). AIMQ: a methodology for information quality assessment. *Information & management, 40*(2), 133-146. doi: http://dx.doi.org/10.1016/S0378-7206(02)00043-5

Leontie, E., Bloom, G., Narahari, B., Simha, R., & Zambreno, J. (2009). Hardware containers for software components: A trusted platform for COTS-based systems. *In Proceeding of 12th IEEE International Conference on Computational Science and Engineering, 2009 CSE'09 held on 29-31 Aug 2009 at the Vancouver, BC* (Vol. 2, pp. 830-836). Canada: IEEE.

Lewis, G., & Morris, E. (2004). From System Requirements to COTS Evaluation Criteria. In R. Kazman & D. Port (Eds.), *COTS-Based Software Systems* (Vol. 2959, pp. 159-168). CA, USA: Springer Berlin Heidelberg.

Li, J. (2006). *Process improvement and risk management in Off-the-Shelf Component-based development*, Unpublished Doctoral Thesis, Faculty of Information Technology, Norwegian University of Science and Technology, Norwegian.

Li, J., Bjørnson, F. O., Conradi, R., & Kampenes, V. B. (2006). An empirical study of variations in COTS-based software development processes in the Norwegian IT industry. *Empirical Software Engineering, 11*(3), 433-461. doi: 10.1007/s10664-006-9005-5

Li, J., Conradi, R., Bunse, C., Torchiano, M., Slyngstad, O., & Morisio, M. (2009). Development with off-the-shelf components: 10 facts. *Software, IEEE, 26*(2), 80-87. doi: 10.1109/MS.2009.33

Li, J., Conradi, R., Slyngstad, O., Bunse, C., Khan, U., Torchiano, M., et al. (2005). An Empirical Study on Off-the-Shelf Component Usage in Industrial Projects. In F. Bomarius & S. Komi-Sirviö (Eds.), *Product Focused Software Process Improvement* (Vol. 3547, pp. 54-68). Oulu, Finland: Springer Berlin Heidelberg.

Li, J., Kvale, A. A., & Conradi, R. (2006). A case study on improving changeability of COTS-based system using aspect-oriented programming. *Journal of Information Science and Engineering, 22*(2), 375-390. Retrieved from http://www.idi.ntnu.no/grupper/su/publ/li/aop-sac12-journalversion.pdf

Lichota, R. W., Vesprini, R. L., & Swanson, B. (1997). PRISM Product Examination Process for component based development. *In Proceedings of 5th International Symposium Assessment of Software Tools and Technologies held on 2-5 Jun 1997 at the Pittsburgh, PA* (pp. 61-69). USA: IEEE.

Lin, H., Lai, A., Ullrich, R., Kuca, M., McClelland, K., Shaffer-Gant, J., et al. (2007). COTS Software Selection Process. *Paper presented at the Proceedings of the 6th International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems held on 26 Feb -2 Mar. 2007 at the Banff, AB* (pp. 114-122). Canada: IEEE.

Lin, H.-Y., Hsu, P.-Y., & Sheen, G.-J. (2007). A fuzzy-based decision-making procedure for data warehouse system selection. *Expert systems with applications, 32*(3), 939-953. doi: 10.1016/j.eswa.2006.01.031

Lincke, R., & Lowe, W. (2007, April). *Compendium of software quality standards and metrics–Version 1.0*. Retrieved on 9 August. 2010 from http://www.arisa.se/compendium/quality-metrics-compendium.pdf

Liu, Y. N. (2010). A Case Study of Evaluating Supplier's Selection Criteria in a Steel Bars Manufacturer. *In Proceeding of IEEE International Conference in Industrial Engineering and Engineering Management, IEEM held on 7-10 Dec 2010 at the Venetian Macao-Resort-Hotel, Macao* (pp. 994-998). China: IEEE.

Lloyd, W. J. (2005). A Common Criteria based approach for COTS component selection. *Journal of Object Technology, 4*(3), 27-34. Retrieved from http://pdf.aminer.org/000/284/530/filtering_cots_components_through_an_improvement_based_process.pdf

Lopez, M. (2000). *An evaluation theory perspective of the Architecture Tradeoff Analysis Method (ATAM)*. (Rep. No. CMU/SEI-2000-TR-012). Carnegie-Mellon University Pittsburgh Pa Software Engineering Inst., USA. Retrieved from DTIC. (ADA387265).

Lopez, M. (2003). Application of an evaluation framework for analyzing the architecture tradeoff analysis method SM. *Journal of Systems and Software, 68*(3), 233-241. doi: http://dx.doi.org/10.1016/S0164-1212(03)00065-7

Lozano-Tello, A., & Gómez-Pérez, A. (2002). BAREMO: how to choose the appropriate software component using the analytic hierarchy process. *In Proceedings of the 14th international conference on Software engineering and knowledge engineering held on July 2002 at the Ischia* (pp. 781-788). Italy: ACM.

Maiden, N., & Ncube, C. (1998). Acquiring COTS software selection requirements. *IEEE software, 15*(2), 46-56. doi: 10.1109/52.663784

Marianos, N. S., Lambrou, M. A., & Spyrou, D. (2011). Evaluating electronic port services for container terminals: the PPA case. *International Journal of Decision Sciences, Risk and Management, 3*(3), 347-368. doi: 10.1504/IJDSRM.2011.046152

Martínez, C. (2008). *Systematic construction of goal-oriented COTS taxonomies*. Unpublished Doctoral Thesis, Faculty of Information Technology, Universitat Politècnica de Catalunya, Spain.

Maxville, V., Armarego, J., & Lam, C. P. (2004). Intelligent component selection. In *Proceedings of the 28th Annual International in Proceeding of the Computer Software and Applications Conference 2004 (COMPSAC) held on 28-30 Sept. 2004* at the *Hong Kong* (pp. 244-249). China: IEEE.

Megas, K., Belli, G., Frakes, W. B., Urbano, J., & Anguswamy, R. (2013). A Study of COTS Integration Projects: Product Characteristics, Organization, and Life Cycle Models. *In*

Proceeding of the 28th ACM SIGAPP Symposium on Applied Computing held on 18-22 Mar 2013 at the Institute of Engineering of the Polytechnic Institute of Coimbra (ISEC-IPC) (pp. 1028-1033). Portugal: ACM.

Mendoza, G. A., Anderson, A. B., & Gertner, G. Z. (2002). Integrating multi-criteria analysis and GIS for land condition assessment: Part 2—Allocation of military training areas. *Journal of Geographic Information and Decision Analysis, 6*(1), 17-30. Retrieved from www.psu.edu

Merola, L. (2006). The COTS software obsolescence threat. *In Proceeding of the 5th International Conference in Commercial-off-the-Shelf (COTS)-Based Software Systems held on 13-16 Feb. 2006 at the San Diego, CA* (pp. 7-13). USA: IEEE.

Mili, A., Chmiel, S. F., Gottumukkala, R., & Zhang, L. (2000). An integrated cost model for software reuse. In Software Engineering, 2000. *In Proceedings of the 2000 International Conference held on 4-11 June 2000 at the Limerick* (pp. 157-166). Limerick, Ireland: IEEE.

Minkiewicz, A. (2005). Six steps to a successful cots implementation. *Crosstalk Journal of Defense Software Engineering*, *18*, 17-22. Retrieved on 23 Jan. 2010 from www.stsc.hill.af.mil

Mohamed, A. S. A. S. (2007). *Decision support for selecting COTS software products based on comprehensive mismatch handling*, Unpublished Doctoral Thesis. Department of Electrical and Computer Engineering, University of Calgary, Alberta.

Mohamed, A., Ruhe, G., & Eberlein, A. (2007). COTS selection: Past, present, and future. *In Proceeding of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07) held on 26 – 29 Mar 2007 at the Tucson, Arizona* (pp. 103-114).USA: IEEE Computer Society.

Mohamed, A., Ruhe, G., & Eberlein, A. (2008). Optimized mismatch resolution for COTS selection. *Software Process Improvement and Practice, 13*(2), 157-169. doi: 10.1002/spip.374.

Mohamed, A., Ruhe, G., & Eberlein, A. (2011). Mismatch handling for COTS selection: a case study. *Journal of Software Maintenance and Evolution: Research and Practice, 23*(3), 145-178. doi: 10.1002/smr.493

Montecillo, M. (2009). *Website Vulnerability Assessment*, (Rep. No. Q4-2009). Enterprise Management Associates (EMA), USA. Retrieved from http://www.enterprisemanagement.com/research/asset-free.php/1617/toc/EMA-Radar-on-Website-Vulnerability-Assessment-Q4-2009-toc

Moody, D. L. (2005). Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data & Knowledge Engineering, 55*(3), 243-276. doi: 10.1016/j.datak.2004.12.005

Moody, D. L., Sindre, G., Brasethvik, T., & Sølvberg, A. (2003). Evaluating the quality of information models: empirical testing of a conceptual model quality framework. *In Proceedings of the 25th International Conference of Software Engineering held on 3 – 10 May 2003 at the Oregon State University, Portland, Oregon* (pp. 295-305). USA: IEEE Computer Society.

Morisio, M., & Torchiano, M. (2002). Definition and Classification of COTS: A Proposal. In J. Dean & A. Gravel (Eds.), *COTS-Based Software Systems* (Vol. 2255, pp. 165-175). FL, USA: Springer Berlin Heidelberg

Morisio, M., & Tsoukias, A. (1997). IusWare: a methodology for the evaluation and selection of software products. *IEE Proceedings in Software Engineering. 144*(3), 162-174. doi: 10.1049/ip-sen:19971350

Morisio, M., Seaman, C., Basili, V., Parra, A., Kraft, S., & Condon, S. (2002). COTS-based software development: Processes and open issues. *Journal of Systems and Software, 61*(3), 189-199. doi: 10.1016/S0164-1212(01)00147-9

Morris, A. T. (2000). COTS Score: an acceptance methodology for COTS software. *In Proceedings of the 19th Digital Avionics Systems Conference DASC hold on 7-13 Oct.2000 at the Philadelphia, PA* (Vol. 1, pp. 4B2/1-4B2/8). USA: IEEE.

Morse, K. L. (2004). Data and metadata requirements for composable mission space environments. *In Proceedings of the 36th Conference on Winter Simulation (WSC '04) held on 5-8 December 2004 at the Washington, D.C* (pp. 271-278). USA: ACM.

Mujeeb-u-Rehman, M., Yang, X., Dong, J., & Abdul Ghafoor, M. (2005). Prioritized selecting COTS vendor in cots-based software development process. *In Proceeding of the Canadian Conference in Electrical and Computer Engineering held on 1-4 May 2005 at the Saskatoon, Sask* (pp. 1939-1945). Canada: IEEE.

Nachmias, F., & Nachmias, D. (1996). *Research methods in the social sciences* (5th Edition). London, UK: Aenold a member of the Hodder Headline Group.

Navarrete, F., Botella, P., & Franch, X. (2007). Reconciling agility and discipline in COTS selection processes. *In Proceeding of the 6th International IEEE Conference in Commercial-off-the-Shelf (COTS)-Based Software Systems (ICCBSS'07) held on 26Febr – 2 Mar 2007 at the Banff, Alta.* (pp. 103-113). Alberta, Canada: IEEE.

Ncube, C., & Dean, J. (2002). The Limitations of Current Decision-Making Techniques in the Procurement of COTS Software Components. In J. Dean & A. Gravel (Eds.), *COTS-Based Software Systems* (Vol. 2255, pp. 176-187). Orlando, USA: Springer Berlin Heidelberg.

Ncube, C., & Maiden, N. A. (1999). PORE: Procurement-oriented requirements engineering method for the component-based systems engineering development paradigm. *In Proceeding of the 2th International Workshop of Component-Based Software Engineering held on 17-18 May 1999 at the Los Angeles, CA* (pp. 130-140). USA: CiteSeerX.

Neubauer, T., & Stummer, C. (2007). Interactive decision support for multi-objective COTS selection. *In Proceeding of the 40th Annual Hawaii International Conference on System Sciences (HICSS 2007) held on 3-6 January 2007 at Waikoloa, HI* (pp. 283b-283b). Hawai, US: IEEE.

Ngai, E., & Chan, E. (2005). Evaluation of knowledge management tools using AHP. *Expert systems with applications, 29*(4), 889-899. doi: 10.1016/j.eswa.2005.06.025

Nguyen, D. K., van den Heuvel, W. J., Papazoglou, M. P., de Castro, V., & Marcos, E. (2009). Gap analysis methodology for business service engineering. *In Proceeding of*

the IEEE Conference in Commerce and Enterprise Computing (CEC'09) held on 20-23 July 2009 at the Vienna (pp. 215-220). Austria: IEEE.

Obeidat, M. A. (2011). Evaluation of Information Technology Vendor Services: An Empirical Study. *International Management Review, 7*(1), 82-88. Retrieved from http://www.usimr.org/IMR-1-2011/v7n111-art10.pdf

Ochs, M., Pfahl, D., Chrobok-Diening, G., & Nothhelfer-Kolb, B. (2001). A method for efficient measurement-based COTS assessment and selection method description and evaluation results. *In Proceedings of the 7th International Software Metrics Symposium (METRICS) held on 04-06 Apr 2001 at the London* (pp. 285-296). UK: IEEE.

Ortega, M., Pérez, M., & Rojas, T. (2003). Construction of a Systemic Quality Model for Evaluating a Software Product. *Software Quality Journal, 11*(3), 219-242. doi: 10.1023/a:1025166710988.

Pallant, J. (2007). *SPSS Survival Manual: A Step by Step Guide to Data Analysis Using SPSS for Windows Version 15* (3th Edition). UK: Open University Press.

Pande, J. (2012). On Some Critical Issues in Component Selection in Component based Software Development. *International Journal of Computer Applications, 46*(4), 44-50. doi: 10.5120/6899-9255.

Pande, J., Garcia, C. J., & Pant, D. (2013). Optimal component selection for component based software development using pliability metric. *ACM SIGSOFT Software Engineering Notes, 38*(1), 1-6. doi: 10.1145/2413038.2413044

Patricia, K., Kathryn, E., & Deborah, A. (2001). A Formal Process for Evaluating COTS Software Products. *IEEE Computer Society, 34*(5), 55-60. Retrieved from http://www.it.iitb.ac.

Pavlovski, C. J., & Zou, J. (2008). Non-functional requirements in business process modeling. *In Proceedings of the 5th Asia-Pacific Conference on Conceptual Modelling (APCCM 2008) held on 22-25 January 2008 at the Wollongong, New South Wales.* (Vol. 79, pp. 103-112). Australia: ACM.

Philips, B. C., & Polen, S. M. (2002). Add Decision Analysis to Your COTS Selection Process. *The Journal of Defense Software Engineering, Software Technology Support Center Crosstalk,* 21-25. Retrieved on 4 April 2011 from http://www.crosstalkonline.org/storage/issue-archives/2002/200204/200204-Phillips.pdf

Pogarcic, I., Francic, M., & Davidovic, V. (2008, October). *Application of AHP Method in Traffic Planning*. Paper presented at the 16th International Symposium on Electronics in Traffic (ISEP 2008) - A Condition for Sustainable Development and Prosperity of A Modern and Safe Transport, Ljubljana Exhibition and Convention Centre, Ljubljana, Slovenia.

Postina, M., Sechyn, I., & Steffens, U. (2009). Gap analysis of application landscapes. *In Proceeding of the 13th Conference in Enterprise Distributed Object Computing Conference Workshops (EDOCW 2009) held on 1-4 Sept. 2009 at the Auckland* (pp. 274-281). Auckland: IEEE.

Rauscher, H. M., Lloyd, F. T., Loftis, D. L., & Twery, M. J. (2000). A practical decision-analysis process for forest ecosystem management. *Computers and Electronics in Agriculture, 27*(1), 195-226. doi: 10.1016/S0168-1699(00)00108-3

Rawashdeh, A., & Matalkah, B. (2006). A new software quality model for evaluating COTS components. *Journal of Computer Science, 2*(4), 373-381.retrieved from http://docsdrive.com/pdfs/sciencepublications/jcssp/2006/373-381.pdf

Rogers, M. R., & Lopez, E. C. (2002). Identifying Critical Cross-Cultural School Psychology Competencies. *Journal of school psychology, 40*(2), 115-141. doi:10.1016/S0022-4405(02)00093-6

Roscoe, J. T. (1975). *Fundamental research statistics for the behavioral sciences*, New York, USA: Holt.

Rowe, G., & Wright, G. (1999). The Delphi technique as a forecasting tool: issues and analysis. *International Journal of Forecasting, 15*(4), 353-375. doi:10.1016/S0169-2070(99)00018-7

Roy, B. (1991). The outranking approach and the foundations of ELECTRE methods. *Theory and Decision, 31*(1), 49-73. doi: 10.1007/BF00134132

Ruhe, G. (2003). Software Engineering Decision Support – A New Paradigm for Learning Software Organizations. In S. Henninger & F. Maurer (Eds.), *Advances in Learning Software Organizations* (Vol. 2640, pp. 104-113). Chicago, IL, USA: Springer Berlin Heidelberg.

Ruth, N. (2008). *A Multi criteria decision making support to software selection*. Unpublished Master's Thesis, Makerere University, Kampala, Uganda.

Saaty, T. L. (1980). *The Analytic Hierarchy Process*. New York, USA: McGraw-Hill.

Saaty, T. L. (2008). Decision making with the analytic hierarchy process. *International Journal of Services Sciences, 1*(1), 83-98. doi: 10.1504/IJSSci.2008.01759

Sankaran, K., Kannabiran, G., & Dominic, P. (2011). Determinants of software quality in COTS products: an exploratory study. *International Journal of Business Information Systems, 8*(1), 4-22. doi: 10.1504/IJBIS.2011.041084

Sargent, R. G. (2012). Verification and validation of simulation models. *Journal of Simulation*, *7*(1), 12-24. doi: 10.1057/jos.2012.20

Sarkar, S. (2012,). Architecture Centric Tradeoff-A Decision Support Method for COTS Selection and Life Cycle Management. *In Proceeding of the Seventh International Conference on Software Engineering Advances (ICSEA 2012) held on 18-23 November 2012 at the Lisbon* (pp. 122-128), Lisbon, Portugal: ThinkMind.

Saunders, M., Lewis, P. and Thornhill, A. (2007). *Research methods for business students* (4th edition). London, UK: Prentice Hall.

Scriven, M. (1991). *Evaluation thesaurus*. Newbury Park, CA: Sage Publications.

Sedigh-Ali, S., Ghafoor, A., & Paul, R. A. (2002). Metrics-based framework for decision making in COTS-based software systems. *In Proceeding of the 7th IEEE International*

Symposium in High Assurance Systems Engineering (HASE 2002) held on 23-25 Oct 2002 at the Tokyo (pp. 243-244). Japan: IEEE.

Seffah, A., Donyaee, M., Kline, R., & Padda, H. (2006). Usability measurement and metrics: A consolidated model. *Software Quality Journal, 14*(2), 159-178. doi: 10.1007/s11219-006-7600-8

Sekaran, U. & Bougie, R. (2010). *Research methods for business: A skill building approach* (5th edition). Chichester, UK: John Willey & Sons Ltd.

Sekaran, U. (2003). *Research methods for business* (4th edition). New York, USA: John Wiley & Sons.

Sen, C. G., & Baracli, H. (2006, May). *A brief literature review of enterprise software evaluation and selection methodologies: A comparison in the context of decision-making methods*. Paper presented at the 5th International Symposium on Intelligent Manufacturing, Department of Industrial Engineering, Sakarya University, Sakarya, Turkey.

Sharifi, M., Van Den Toorn, W., Rico, A., & Emmanuel, M. (2003). Application of GIS and multi-criteria evaluation in locating sustainable boundary between the Tunari National Park and Cochabamba City (Bolivia). *Journal of Multi-Criteria Decision Analysis, 11*(3), 151-164. doi: 10.1002/mcda.323

Sharma, A., Kumar, R., & Grover, P. (2007). Managing Component-Based Systems with Reusable Components. *International Journal of Computer Science and Security, 1*(2), 60. Retrieved from http://www.cscjournals.org/csc/manuscript/Journals/IJCSS/Volume1/Issue2/IJCSS-16.pdf

Sharma, A., Kumar, R., & Grover, P. S. (2008). Estimation of quality for software components: an empirical approach. *SIGSOFT Software. Eng. Notes, 33*(6), 1-10. doi: 10.1145/1449603.1449613

Sheng, J., & Wang, B. (2008). Evaluating COTS Components Using Gap Analysis. *In Proceeding of the 9th International Conference for Young Computer Scientists (ICYCS 2008) held on 18-21 Nov. 2008 at the Hunan* (pp. 1248-1253). Hunan, China: IEEE.

Simão, R. S., & Belchior, A. (2003). Quality Characteristics for Software Components: Hierarchy and Quality Guides. In A. Cechich, M. Piattini & A. Vallecillo (Eds.), *Component-Based Software Quality* (Vol. 2693, pp. 184-206). Brazil: Springer Berlin Heidelberg.

Singh, M., Khan, I., & Grover, S. (2011). Application of AHP in Measuring and Comparing Quality of Manufacturing Industries. *International Journal Of Multidisciplinary Sciences And Engineering, 2*(3), 6-13. Retrieved from http://www.ijmse.org/Volume2/Issue3/paper2.pdf

Skramstad, T. (2005, Sept). *Assessment of safety critical systems with COTS software and software of uncertain pedigree (SOUP)*. Paper presented at the ERCIM Workshop on Dependable Systems, ERCIM Conference on Dependable Software intensive embedded Systems, Italy.

Solberg H. and Dahl K. M.,(2001). *COTS Software Evaluation and Integration Issues*. (Rep. No.SIF8094), Norwegian University of Technology and Science, Software Engineering Project, Trondheim, Norway. Retrieved from www.idi.ntnu.no

Sommerville, I. (2001).*Sofware Engineering* (6th edition). UK: Addison- Wesley.

Sommerville, I. (2004). *Software Engineering* (7th Edition). UK: Pearson Education.

Sommerville, I. (2011). *Software Engineering* (9th Edition). UK: Addison-Wesley.

Soni, G., & Kodali, R. (2010). A multi-criteria decision model for the selection of supply chain management software. *International Journal of Services and Operations Management, 7*(4), 501-533. doi: 10.1504/IJSOM.2010.03571

Sun, C.-C. (2010). A performance evaluation model by integrating fuzzy AHP and fuzzy TOPSIS methods. *Expert systems with applications, 37*(12), 7745-7754. doi: 10.1016/j.eswa.2010.04.066.

Tam, F. (2012). *Definitions, Concepts, and Principles*. Service Availability, 1-21. Retrieved on 17 August 2009 from http://content.schweitzer-online.de/ static/content/ catalog/newbooks/978/111/ 995/ 9781119954088/9781119954088_Excerpt_001.pdf

Tektas, A., & Aytekin, A. (2011). Supplier selection in the international environment: a comparative case of a Turkish and an Australian company. *IBIMA Business Review, 2011*(1), 1-14. doi: 10.5171/2011.598845

Thacker, B. H., Anderson, M. C., Senseny, P. E., & Rodriguez, E. A. (2006). The role of nondeterminism in model verification and validation. *International Journal of Materials and Product Technology, 25*(1), 144-163. Retrieved from http://inderscience.metapress.com/content/eex43vkgr5ylu0j5/

Thapar, S., Singh, P., & Rani, S. (2012). Challenges to the Development of Standard Software Quality Model. *International Journal of Computer Applications, 49*(10), 1-7. doi: 10.5120/7660-0765

Torchiano, M. (2001). *Selected Literature on COTS products.* USA: NASA Electronic Parts and Packaging (NEPP). Retrieved on 25 March 2009 from http://nepp.nasa.gov/DocUploads/6ABAC4DFB96C430185E6C00ECD6399B2/Selected%20Literature%20on%20COTS%20products.pdf

Torchiano, M., & Morisio, M. (2004). Overlooked aspects of COTS-based development. *IEEE software, 21*(2), 88-93. doi: 10.1109/MS.2004.1270770

Torchiano, M., Jaccheri, L., Sørensen, C. F., & Wang, A. I. (2002). COTS products characterization. *In Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE 2002) held on 15-19 July 2002 at the Ischia* (pp. 335-338). Italy: ACM.

Tran, V., Liu, D. B., & Hummel, B. (1997). Component-based systems development: challenges and lessons learned. *In Proceedings of 8th IEEE International Workshop of Software Technology and Engineering Practice held on 14-18 Jul 1997 at the incorporating Computer Aided Software Engineering* (pp. 452-462). London, UK: IEEE.

Trochim, W. M. (2006). *Qualitative measures. Research Measures Knowledge Base*, (pp.361-9433). Retrieved on 13 May 2010 from http://www.socialresearchmethods.net/kb/qualval.php.

Tsai, W.-H., Chou, W.-C., & Lai, C.-W. (2010). An effective evaluation model and improvement analysis for national park websites: A case study of Taiwan. *Tourism Management, 31*(6), 936-952. doi: 10.1016/j.tourman.2010.01.016

Turksoy, H., Uysal, M., Cetinkaya, O., Malas, A., Akcaoglu, I., & Okur, Y. (2012). Building a layered enterprise architecture using COTS products for NATO air command & control information services. *In Proceeding of the Communications and Information Systems Conference (MCC) held on 8-9 Oct. 2012 at the Gdansk* (pp. 1-6). Gdansk, Poland: IEEE.

Ünal, C., & Güner, M. G. (2009). Selection of ERP suppliers using AHP tools in the clothing industry. *International Journal of Clothing Science and Technology, 21*(4), 239-251. doi: 10.1108/09556220910959990

Ulkuniemi, P., & Seppanen, V. (2004). COTS component acquisition in an emerging market. *IEEE software, 21*(6), 76-82. doi: 10.1109/MS.2004.38.

Vaidya, O. S., & Kumar, S. (2006). Analytic hierarchy process: An overview of applications. *European Journal of Operational Research, 169*(1), 1-29. doi: http://dx.doi.org/10.1016/j.ejor.2004.04.028

Vega, J. P. C. (2006, February). Supporting organizational induction and goals alignment for COTS components selection by means of i*. *In Proceeding of the 5th International Conference in Commercial-off-the-Shelf (COTS)-Based Software Systems held on 13-16 Feb. 2006 at the Orlando* (pp. 8-pp).Florida, USA: IEEE.

Vijayalakshmi, K., Ramaraj, N., & Amuthakkannan, R. (2008). Improvement of component selection process using genetic algorithm for component-based software development. *International Journal of Information Systems and Change Management, 3*(1), 63-80. doi: 10.1504/IJISCM.2008.019289

Vitharana, P., Zahedi, F., & Jain, H. (2003). Knowledge-based repository scheme for storing and retrieving business components: a theoretical design and an empirical analysis. *IEEE transactions on Software Engineering, 29*(5), 649-664. doi: 10.1109/TSE.2003.1214328

Wallnau, K. C., Hissam, S., & Seacord, R. C. (2002). *Building systems from commercial components*. USA: Addison-Wesley.

Wallnau, K., Carney, D., & Pollak, B. (1998). How COTS software affects the design of COTS-intensive systems. *SEI interactive, 6*, 98. Retrieved from http://interactive.sei.cmu.edu/Features/1998/june/cotssoftware/CotsSoftware.htm

Wanyama, T., & Far, B. (2005). A Multi-Agent Framework for Conflict Analysis and Negotiation: Case of COTS Selection. *IEICE transactions on information and systems, 88*(9), 2047-2058. Retrieved from http://search.ieice.org/bin/summary.php?id=e88-d_9_2047

Wanyama, T., & Far, B. (2008). An Empirical Study to Compare Three Methods for Selecting Cots Software Components. *International Journal of Computing and ICT Research, 2*, 34-46. Retrieved 25 July 2009 from

www.siia.net/software/pubs/growth_software03.pdf

Wanyama, T., & Lumala, A. F. N. (2007). Decision Support for the Selection of COTS. *International Journal of Computing and ICT Research, 1*(2), 33-43. Retrieved from http://www.ijcir.org/volume1-number/article4.pdf.

Wei, C.-C., Chien, C.-F., & Wang, M.-J. J. (2005). An AHP-based approach to ERP system selection. *International Journal of Production Economics, 96*(1), 47-62. doi: 10.1016/j.ijpe.2004.03.004

Wile, D., Balzer, R., Goldman, N., Tallis, M., Egyed, A., & Hollebeek, T. (2010). Adapting COTS products. *In Proceeding of the Software Maintenance (ICSM) in IEEE International Conference held on 12-18 Sept. 2010 at the Timisoara* (pp. 1-9). Romania: IEEE.

Yahaya, J. (2006). *The development of software certification model based on product quality approach*. Unpublished Doctoral Thesis. Faculty of Information Science and Technology, University of Kebangsaan Malaysia. Malaysia.

Yanes, N., Sassi, S. B., & Ghezala, H. H. B. (2012, Apr). *A Theoretical and Empirical Evaluation of Software Component Search Engines, Semantic Search Engines and Google Search Engine in the Context of COTS-Based Development*. arXiv preprint arXiv:1204.2079, Cornell University Library, NY,USA. Retrieved on 7 Oct. 2012 from http://arxiv.org/abs/1204.2079

Yang, L., Dang, Z., & Fischer, T. R. (2011). Information gain of black-box testing. *Formal aspects of computing, 23*(4), 513-539. doi: 10.1007/s00165-011-0175-6

Yang, Y., Bhuta, J., Boehm, B., & Port, D. (2005). Value-based processes for COTS-based applications. *IEEE software, 22*(4), 54-62. doi: 10.1109/MS.2005.112

Ye, F., & Kelly, T. (2004). COTS Product Selection for Safety-Critical Systems. In R. Kazman & D. Port (Eds.), *COTS-Based Software Systems* (Vol. 2959, pp. 53-62). CA, USA: Springer Berlin Heidelberg.

Yin, R. (2003). *Case Study Research Design and Methods* (3rd edition). London, UK: Sage.

Yu, E. (1997). Towards *modeling and reasoning support for early-phase requirements engineering*. RE '97: Proceedings of the 3Th IEEE International Symposium on Requirements Engineering, page 226.

Zarour, M. (2009). *Methods to evaluate lightweight software process assessment methods based on evaluation theory and engineering design principles*. Unpublished Doctoral Thesis. Computer Science Department, Du Quebec University, Canada.